



# LABKIT BOOKLET







# LABKIT BOOKLET



**Published by Ellinogermaniki Agogi**

**Authors:**

Georgios Mylonas, Dimitrios Amaxilatis, Lidia Pocero, Iraklis Markelis, Stylianos Tsampas  
*Computer Technology Institute and Press “Diophantus”*  
Giovanni Cuffaro, Federica Paganelli  
*National Inter-University Consortium for Telecommunications*

**Editor:**

Pavlos Koulouris  
*Ellinogermaniki Agogi*

**Artwork:**

Jörg Hofstätter, Peter Matis, Maria Raser  
*ovos media*

Anna Mavroeidi  
*Ellinogermaniki Agogi*



Funded by the Horizon 2020  
Framework Programme of the  
European Union

This project has received funding from the European Union's Horizon 2020 Programme under grant agreement no. 696029.

The content of this publication does not reflect the official opinion of the European Union. Responsibility for the information and views expressed in the document lies entirely with the authors.



© 2019

Reproduction or translation of any part of this work without the written permission of the copyright owners is unlawful. Requests for permission or further information should be addressed to Computer Technology Institute and Press “Diophantus”, Patras, Greece, and to Ellinogermaniki Agogi, Athens, Greece.

Printed by EPINOIA S.A.



# CONTENTS

5

## CHAPTER 1

5 | Introduction

9

## CHAPTER 2

9 | Description of the Lab Kit's main software and hardware components

15

## CHAPTER 3: Preparatory Activities

15 | Computer Programming  
STUDENTS: preparatory activity Level Beginner Computer Programming - Preparatory Activities  
16 | TEACHERS: 2nd preparatory activity Level Beginner Labs GAIA - Introduction

19

## CHAPTER 4: Introduction

19 | GrovePi+ Demonstration  
STUDENTS: 1st WorkSheet Level Beginner Introduction  
21 | GrovePi+ Presentation  
TEACHERS: 1st WorkSheet Level Beginner Introduction  
22 | Circuit Elements  
STUDENTS: 2nd WorkSheet Level Beginner Introduction  
27 | TEACHERS: 2nd WorkSheet Level Beginner Introduction  
28 | Control Panel  
STUDENTS: 3rd WorkSheet Level Beginner Introduction  
31 | TEACHERS: 3rd WorkSheet Level Beginner Introduction

33

## CHAPTER 5: Lighting

33 | Light Sensors  
STUDENTS: 1st WorkSheet Level Beginner Lighting  
35 | Demonstration of GrovePi+  
TEACHERS: 1st WorkSheet Level Beginner Lighting  
36 | Light control  
STUDENTS: 2nd WorkSheet Level Beginner Lighting  
38 | TEACHERS: 2nd WorkSheet Level Beginner Lighting  
39 | Record history  
STUDENTS: 3rd WorkSheet Level Beginner Lighting  
41 | TEACHERS: 3rd WorkSheet Level Beginner Lighting

43

## CHAPTER 6: Temperature

43 | Monitoring Temperature - Humidity  
STUDENTS: 1st WorkSheet Level Beginner Temperature  
44 | TEACHERS: 1st WorkSheet Level Beginner Temperature  
45 | Heat Index & Discomfort Index  
STUDENTS: 2nd WorkSheet Level Beginner Temperature  
47 | TEACHERS: 2nd WorkSheet Level Beginner Temperature  
48 | Historical data  
STUDENTS: 3rd WorkSheet Level Beginner Temperature  
50 | TEACHERS: 3rd WorkSheet Level Beginner Temperature  
51 | External & internal conditions  
STUDENTS: 4th WorkSheet Level Beginner Temperature  
TEACHERS: 4th WorkSheet Level Beginner Temperature

55

## CHAPTER 7: Electrical Current

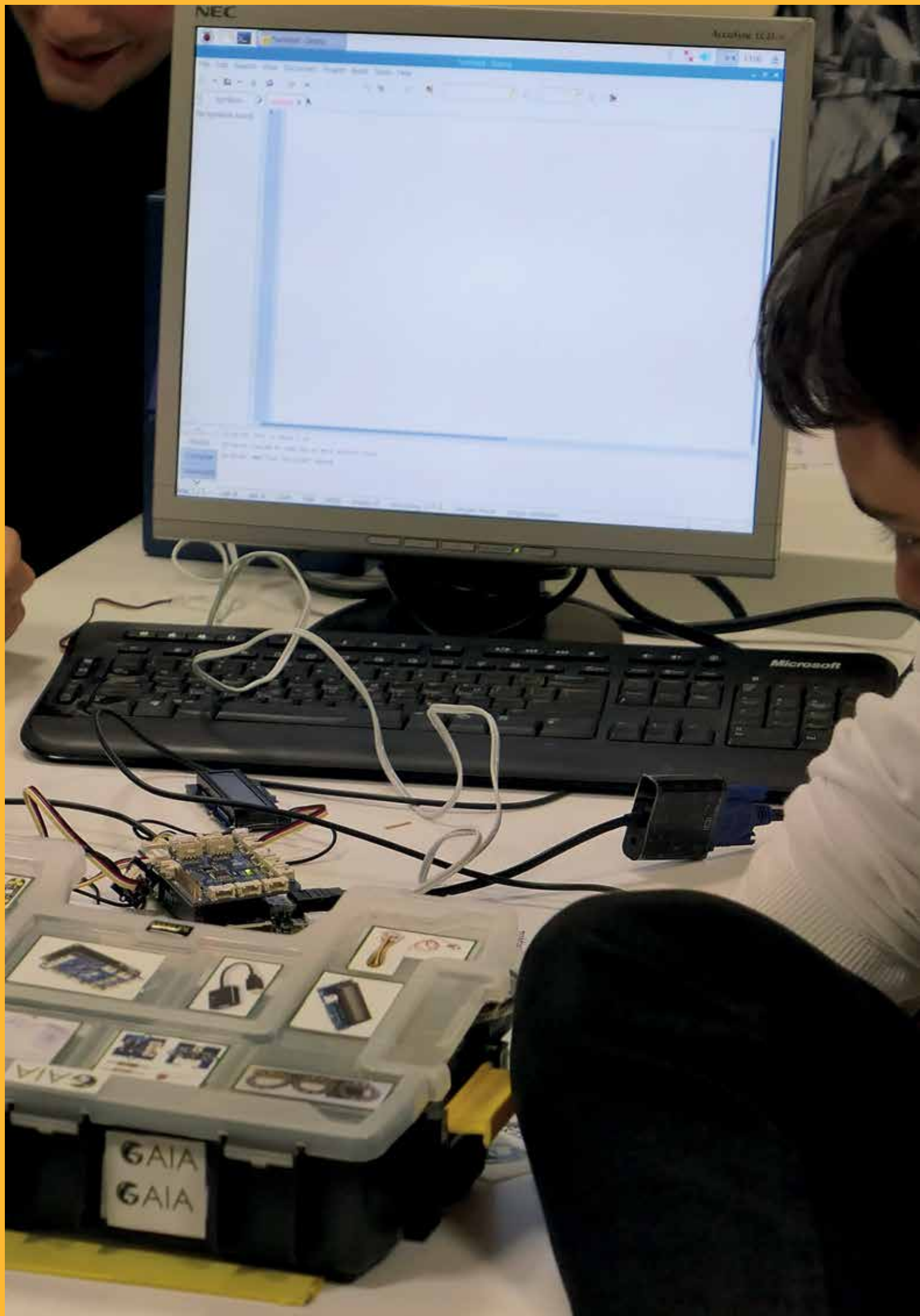
55 | Power usage monitoring  
STUDENTS: 1st WorkSheet Level Beginner Electrical Current  
58 | TEACHERS: 1st WorkSheet Level Beginner Electrical Current  
59 | Daily Historical Data  
STUDENTS: 2nd WorkSheet Level Beginner Electrical Current  
61 | TEACHERS: 2nd WorkSheet Level Beginner Electrical Current  
62 | Historical Data during Recesses  
STUDENTS: 3rd WorkSheet Level Beginner Electrical Current  
63 | TEACHERS: 3rd WorkSheet Level Beginner Electrical Current

65

## CHAPTER 8

65 | Gaia Nodes







# CHAPTER 1

## Introduction

The GAIA (Green Awareness In Action)<sup>1</sup> project has developed an Internet of Things (IoT) platform that combines sensing, Web-based tools and playful learning elements, aimed at the educational community. Its goal is to increase awareness about energy savings and sustainability, based on real-world sensor data produced inside the actual school buildings where students and teachers live and work, while also lead towards behavior change in terms of energy efficiency.

In this book, we present the *GAIA educational lab kit*: a set of activities that utilize the IoT infrastructure of the project developed over open-source technologies. The target audience for this book is the educational community, as well as makers and researchers who would like to experiment with IoT hardware and software in the context of energy and sustainability. As such, the focus here is on presenting a series of educational lab activities to familiarize students with concepts related to energy consumption, sustainability and building monitoring, using popular IoT technologies. These lab activities have been produced throughout the course of GAIA, and have been put to practice at a number of schools participating in the project between February 2018 and April 2019. They were also designed to be replicable for any schools that wish to combine science/technology classes with a twist on energy efficiency and sustainability. In other words, they are not strictly limited to the context and specific technological implementation of GAIA, but can be adopted to fit other contexts that utilize similar implementations and data, since they use open source technologies.

With respect to the overall concept behind the project, GAIA utilizes a number of IoT installations in school buildings in Greece, Italy and Sweden. In order to affect the behavior of students and teachers in terms of energy consumption and achieve sustainable results, GAIA utilizes a loop-based approach focused around three pillars: *raise awareness*, *support action* and *foster engagement*. Each school installation consists of a multitude of IoT nodes, which communicate with a cloud infrastructure via a gateway device. Such nodes comprise multiple sensing endpoints, while the gateway nodes coordinate communication and enable interaction with cloud-based services.

Reflecting these design decisions, the hardware design of the IoT nodes utilized in GAIA uses widely – available hardware components, paired with some custom designs, such as custom printed circuit boards for interconnecting sensors to microcontrollers. Almost all available infrastructure is based on Arduino-compatible or Raspberry Pi components. GAIA releases the specifications for this infrastructure and

related results as open-source, whenever possible. For this purpose, related material is available at the project's GitHub repository<sup>2</sup>.

This book contains in two parts, meant to differentiate between lab activities that rely on set of different technologies. The first part present lab activities that were developed using popular hardware IoT technologies, paired with software and activity templates developed within the project. The second part of this book presents a smaller set of lab activities that rely on the use of node-RED, a popular software platform for implementing IoT-related activities. In both cases, the material included in this book is meant to utilize environmental and power consumption data and use them in the spirit of the GAIA project.

### Chapters 1-7 - The GAIA Educational Lab Kit

In Chapters 1-7 of this book, we present in detail the core activities of the GAIA Educational Lab Kit. It aims to help students study some aspects of energy consumption and monitoring conditions inside buildings using a “hands-on” approach, in which they get to use IoT components and electronics, examine data from their school building and go through the peculiarities of consuming energy, or how the building behaves inside classrooms in terms of environmental parameters, among other things. Essentially, the focus in the lab kit activities is on enabling students to “map” what is actually happening inside their school building in real time. In addition, together with their teachers, students can conduct simple “missions” inside the different parts of their buildings, indoors and outdoors, detecting and pinpointing specific energy-efficiency-related issues in rooms or devices.

The Lab Kit includes IoT devices, commercial or GAIA-designed, together with other hardware components that can be used to assemble custom electrical circuits, which can enable familiarization with electronics, using common components, such as LEDs, resistors, switches, etc. The goal is to monitor real-world parameters, in a way similar to the fixed IoT infrastructure installed inside GAIA school buildings, but in a more “personalized” and direct manner.

With respect to wiring components, students can “draw” the interconnections required using conductive ink. In this manner, we can avoid the extensive use of cables and breadboards, simplifying the overall activity and save time. The kit utilizes Raspberry GrovePi “hats” that have standardized connectors for input/output interfaces. We use electronic components that on the one edge connect to these “hats”

<sup>1</sup> <http://gaia-project.eu>

<sup>2</sup> <https://github.com/GAIA-project>





and on the other carry magnets, in order to enable easy placement on top of a metal surface. The use of magnets prevents the components from slipping and moving around during the lab kit activity. We also utilize printed school building floorplans (see e.g., the ones included in the annex of this document). The floorplans are meant to be placed on top of a metal surface, so that magnets are kept firmly connected during the lab activities.

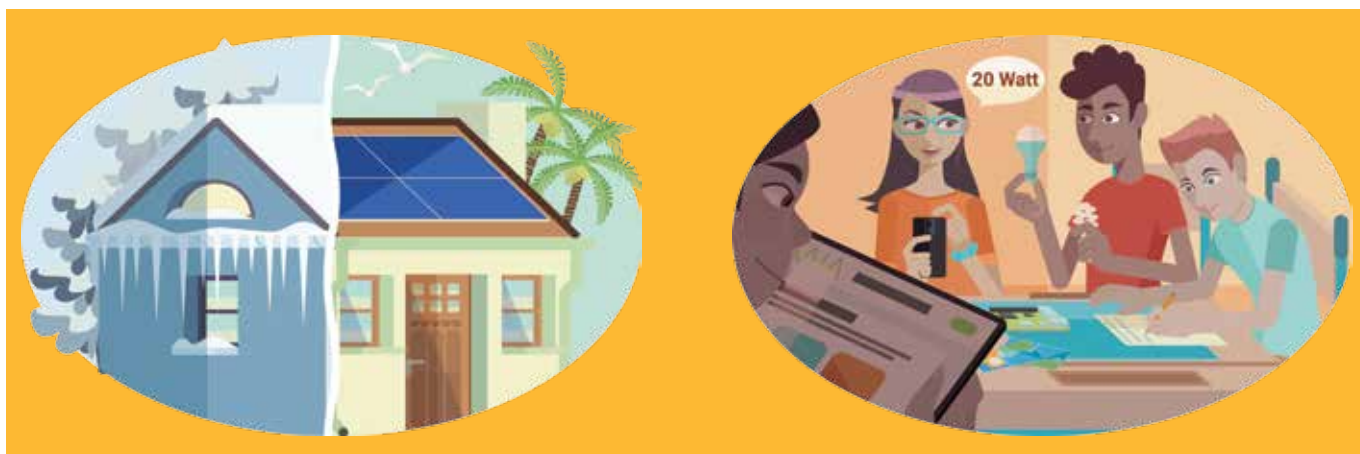
Regarding the actual bill of materials for the Lab Kit (explained in more detail in subsequent chapters of this document), we utilize the following components:

- Raspberry Pi model 3B or 3B+, as the device handling the main computing and networking duties for the lab activities.
- Conductive ink markers, for sketching out wire paths onto paper.
- Electronic components, such as LCD screens, resistors and switches. We use components with magnetic clip attachments, due to ease of use and safety.
- Open-source GrovePi sensors with standardized interfaces for connecting to the Raspberry Pi.
- Custom electronic boards, which ease interfacing with GAIA's cloud services and visualization of real-time data used during the lab activities, such as a LED ring board to visualize energy consumption.

Regarding the software part of the Lab Kit, we use the Raspbian Linux distribution, while the activities are based on a number of Python scripts. The activities use the default options for Python provided by Raspbian. We use the standardized Web interfaces of GAIA for communicating with the system for real-time data.

GAIA has prepared a series of lab activities, covering aspects of energy consumption and efficiency inside school buildings. The thematic list covered is the following: a) Energy consumption in our school, b) Lighting inside school buildings, c) Temperature, Humidity and Thermal Comfort, d) Devices and Energy efficiency, and e) Energy Inspectors - The energy footprint of our building. The last two activities are based on using additional smartphone apps and equipment, and are not covered in this book.

Regarding the activities included in this book, we provide guides for each one of them, in the form of lesson plans for the educators and activity guides for the students. In the description of each activity, we include the title of the subject, the necessary cognitive background for the teams (theoretical and practical) and a short description of the tasks to be completed (goals). One set of material concerns the educators, identifying the educational target for each activity, the methods used, as well as a schedule for the proposed lab activity. Another set of material addresses the students' part, giving specific instructions on how to perform the envisioned







activities, explaining how to interconnect sensors and electronic components, and how to execute the Python scripts provided by the project. Difficulty levels are also indicated in the material, with more complex challenges such as coding questions and exercises are available for e.g., high school, or more advanced students.

## Chapter 8 - The GAIA-Node – A node-RED plugin

Node-RED<sup>3</sup> is a visual tool for wiring the Internet of Things, but it can also be used for other types of applications to assemble flows of services in an easy manner. It is a popular tool available as open source software, used for the development of Internet of Things applications, or as a general-purpose event-processing engine. The GAIA-Node plugin for Node-RED is a set of “nodes” that allow to interact with the GAIA platform to access available resources, such as power consumption measurements from the available infrastructure. Mainly the nodes handle the implementation details related to querying GAIA’s platform and retrieving the measurements gathered by the sensors in GAIA’s schools, thus greatly simplifying interaction with the project.

In Chapter 8 of this book, we present a set of node-RED based activities designed within GAIA, aimed specifically

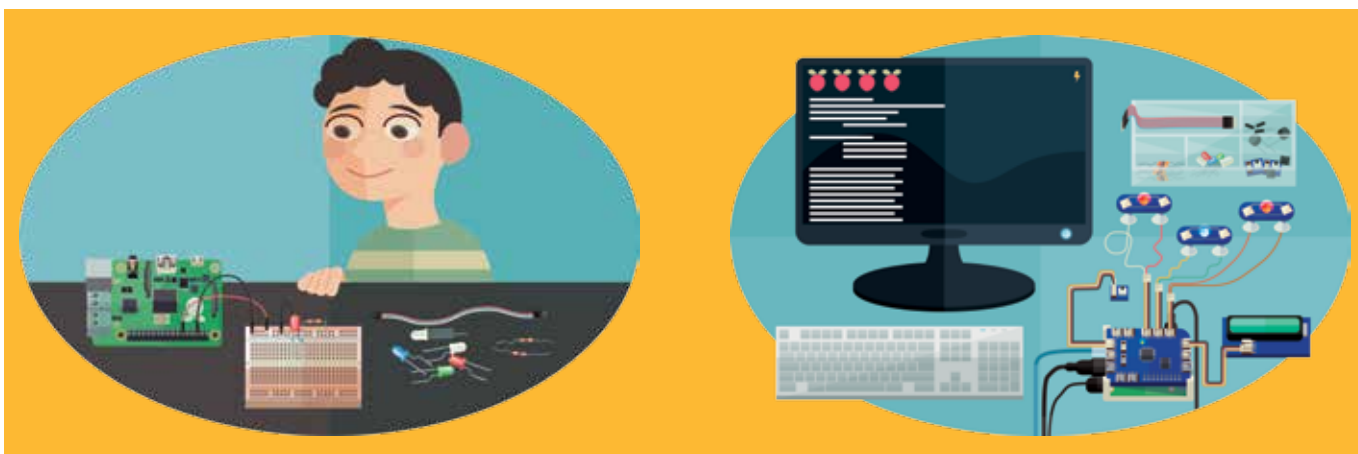
<sup>3</sup> <https://nodered.org/>

at schools. A set of video tutorials has also been produced to support teachers in instructing students for these specific lab activities, which can be found on YouTube and the project’s website. The covered topics include node-RED basics, reading values from a sensor, or creating a Web dashboard to display GAIA readings.

## The GAIA GitHub repository

The repository of the GAIA project on GitHub<sup>4</sup> is a useful companion to the lab activities, containing the actual code utilized during the implementation of the activities, as well as useful information and additional details especially regarding the node-RED based lab activities. In this context, the readers are advised to check for any additions, or corrections, that may be available the future through this repository.

<sup>4</sup> <https://github.com/GAIA-project/Gaia-Workshop>









## CHAPTER 2

# Description of the Lab Kit's main software and hardware components

### The GAIA lab kit briefcase

For the purpose of sharing and moving GAIA lab kit components between the various schools that participated in the project, we assembled a number of identical sets of equipment, and store these component sets inside a small briefcase. This makes it easy to move the equipment required for the lab activities from one school to the other, as well as simplifying the sharing of the components between various student teams. You can see an example of the GAIA lab kit “briefcase” in the following figure.



In terms of components, each briefcase contained the items included in the following table, with their number indicated inside the parentheses.

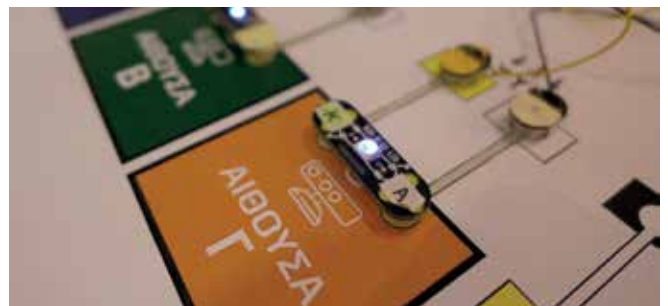
LED (3)	Raspberry Pi (1)
Switch (1)	Grove Pi (1)
2-pin receptor (1)	LED Rings (1)
10K $\Omega$ resistor (2)	LCD Screen (1)
470K $\Omega$ resistor (2)	Power Supply (1)
Photoresistor (1)	USB cable (1)
Button (2)	HDMI to VGA adapter (1)
Photodiode (1)	2-pin cable (3)
Conductive ink pen (1)	3-pin cable (1)

As mentioned in the introduction of this book, one goal for the activities was to make it easy for students to related to the readings used during the implementation of the lab

activities. This is achieved on a first level by using a floor-plan of their school (referred to from now on as the “map”) as a surface to place electronic components and devices, essentially assembling a small-scale interactive installation. In addition, many of the electronic components used in the kit (as described in the rest of this chapter) utilize magnetic connectors. In order to combine these two concepts, we utilize a metal plate underneath the printed floorplan, so that the components can attach to the plate magnetically and be stable throughout the duration of the activities, as seen in the following photo (the metal plate is the yellow surface underneath the floormap).



While we use cables to connect sensors and electronic components with the Raspberry Pi, the rest of the connections are made using a conductive ink pen, essentially substituting wires. Using this approach, we minimize the probability of making a short-circuit or a wrong connection.



### The Grove family of sensors and electronic components

For the implementation of GAIA's lab activities described in this book, we chose the Grove line of sensors and com-



ponents, a popular family of IoT software and hardware components, that follows an open-source approach. The creators of this line of components follow a “manufacturing as a service” model, which enables other developers to utilize existing designs and expand on them. An example of this approach is the GrovePi module used in the activities for interfacing Grove sensors with a Raspberry Pi, whose software and hardware implementation is available on GitHub<sup>1</sup> under the MIT open source license.

## The Circuit Scribe family of components

Circuit Scribe offers a line of electronic components and conductive ink products that are meant to be used by students in an educational context, helping them to learn the basics of electronics while also being safe to use inside a classroom. We chose to use them in GAIA’s lab activities due to their clever design and safety features, originating from the extensive use of magnets as. Since the lab activities use only standard electronic components such as resistors and LEDs, you can always substitute the Circuit Scribe components with other similar items, or even build your own.

## Raspberry Pi

The Raspberry Pi is essentially a small computer in credit card size, designed to enable all sorts of people to experiment with electronics and also learn the basics of computer science. It comes in a single-board form, including all ports required to interface with all sorts of devices. The ports used for the GAIA lab activities are the HDMI port for computer screen interfaces, and USB ports for a keyboard and mouse. It also has a micro USB port used to power it, as well as a microSD card slot with a memory card usually containing Raspbian, a Linux distribution designed specifically for the Raspberry Pi.



## Raspberry Pi Power Supply



The Power Supply provides Raspberry Pi with the power it needs to operate through a micro USB port. The power supply provides a voltage of 5.1V and a current of 2.5A. It is recommended to connect the

power supply after all other components have been connected in order to avoid short circuits during assembly.

## Keyboard and mouse

Raspberry Pi only supports the USB interface, so the keyboard and mouse you use should use the same interface.



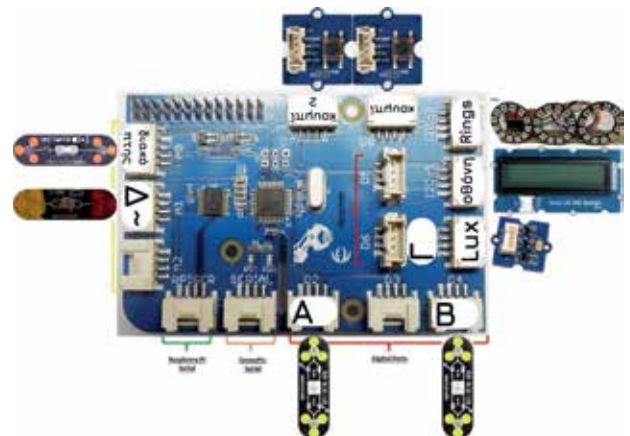
## HDMI to VGA adapter (optional)

The HDMI to VGA Converter is used to connect Raspberry Pi to older computer screens (often found in many schools). The male metallic connection is connected to the Raspberry Pi, while the blue female port connected to the display cable (usually blue, sometimes black).



## Grove Pi interconnection board

The Grove Pi<sup>1</sup> is an expansion board designed specifically for Raspberry Pi, providing interface ports to connect to the family of Grove electronic components, which are utilized in the GAIA kit. In the following figure, you can see an example of various elements attached to it, such as the Grove LCD Display, the Buttons, the LED Ring, the Brightness Sensor, and the Map Component Connection Cables. The map components include 2-color LED elements, a switch element and two recessed connection elements, on top of which resistors and photoresistors can be mounted.



## Grove LCD screen

The Grove LCD screen is used to display the information provided by sensors, as well as values calculated during code execution. The screen connects through the I2C bus and supports Latin



<sup>1</sup> <https://github.com/DexterInd/GrovePi>



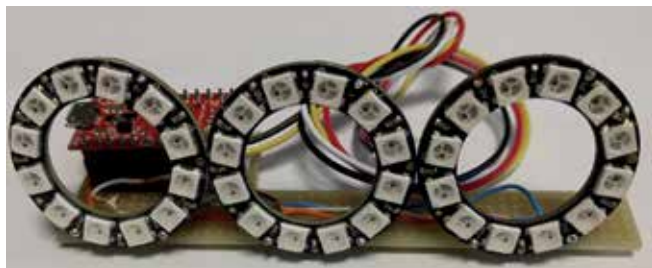
character sets. It can display 16 characters in each of its 2 lines. It can also color its backlight based on the RGB standard. The color of the screen in the lab activities is used to indicate the room for which results are displayed, using the purple, orange and green color. When displaying in grey color, the screen is used to display general information.



### Grove Buttons

Each Button provides instant turn on/off, with the key recovering after being released. It emits a HIGH signal (1) when pressed and LOW (0) when released.

### Custom LED rings component



The LED rings<sup>2</sup> component is used to display the measured quantities in a way that facilitates comparison between them. They communicate with the Grove Pi through an I2C bus and consist of an Arduino Pro Micro microcontroller and 3 rings of twelve LEDs. Each LED can be colored based on the RGB standard. The microcontroller software supports a value from 0 (0x0) to 15 (0xF) for each ring, which must be in character format of the ASCII template. Communication is done using four alphanumeric characters, the three first being the number of LEDs open in each ring and the fourth one by a new line character. The microcontroller software does not support ring color change, which is fixed in purple, orange, and green colors.

### Grove Digital Luminosity Sensor

The Digital Luminosity Sensor is based on the TSL2561 chip for converting light intensity into a digital signal and communicating with the Grove Pi via the I2C bus. In contrast to a typical



analog luminosity sensor, this sensor allows to use the visible spectrum. You can switch between 3 modes: infrared, full spectrum and visible spectrum. When used in visible spectrum mode, it returns readings representative of human sight.

### 2-terminal Cables



We use 2-terminal Cables to interface ports A, B and C on the Grove Pi with LEDs. The magnetic contacts of the cables are colored in white and yellow. To properly display the data, the terminals should be linked based on their color in the corresponding squares above the map halls (you can see an example in the annex).

### Triple connector cables



The 3-pin Terminal Cable is used to connect the Switch with the switch circuit on the map. The terminals are red, yellow and black. Red is for power, yellow carries the Grove Pi signal and black is the neutral. When the switch cables are connected to the circuit, you should be careful so that the black and red terminals do not short-circuit, as this can cause the Raspberry Pi to restart and even potentially damage it. Therefore, you should first place the magnetic parts on the map and then connect to the Grove Pi port.

### Conductive ink

Conductive Ink is used to interconnect the map circuit. This ink has the special feature of high electrical conductivity, allowing easy cre-



<sup>2</sup> <https://github.com/GAIA-project/Gaia-Workshop/tree/master/firmware>



ation of circuits on paper. You will need to draw such circuits in the shapes to place the Cables, the 2-Color LEDs, the Switch and the Two Terminal Connectors.

## 2-color LEDs



2-color LEDs are used to represent the condition of the room that each exercise watches

es on the map. LEDs are usually of one color and lead the electric current only in one direction, as does a diode. Their terminals are colored yellow and white and correspond to the color of the connection squares in the halls and cable terminals. The lab's LEDs can drive the current in two directions and change color depending on the direction. When the LED goes from the yellow terminal to the white, ie the yellow terminal is HIGH (1) and the white LOW (0), the LED emits blue light. In the opposite case, ie when the yellow terminal is LOW (0) and the white HIGH (1), the current goes off and the LED emits red light. If both terminals are LOW (0) or HIGH (1) the LED remains off.

## Switches



The Switch is used to select different functions in the lab programs as a complement to the Buttons. The switch has two

terminals, one yellow and one red. It is used with the Three Terminal Cable and is connected to the terminals with the corresponding colors. The red is connected to the trend and the yellow one at the signal input of Grove Pi. The switch is closed to the yellow terminal and open to red.

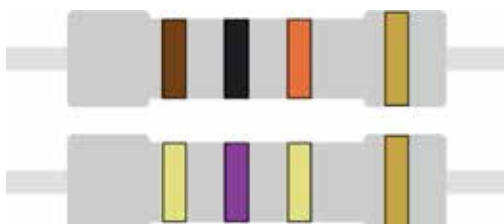
## 2-pin connectors



2-pin Connectors are used to mount Resistors and Photoreceptors on the map. They can accept any other element of an electrical circuit having two terminals. There are

two such kits in the kit. The first one has its magnetic elements in yellow and black, and has a Resistor connected to it. The second one has colors yellow and red and has a Photoreceptor attached to it.

## Resistors



A Resistor is a passive electric element with two terminals

that applies the electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, divide voltages, activate active elements, and terminate transmission lines, among other uses. The resistance is measured in Ohm ( $\Omega$ ) and the resistance values are recognized based on the colors of the lines seen in their body. There are two 10k ohms (black-orange-gold) and a 470k Ohms (yellow-purple-yellow-gold) resistor in the kit.



## Photoresistor

Photoresistors are variable light-controlled resistors. The resistance of a photoresistor is reduced by the increasing intensity of the incident light, i.e., it exhibits photoconductivity.

A photo resistor can be applied to light intensity detector circuits and light-switched circuit-breaker circuits. In the dark, a photoresistor can have a resistance as high as several megohms ( $M\Omega$ ), whereas in environments with plenty of light it can have a resistance as low as a few hundred Ohms. The resulting free electrons (and their respective holes) drive the electricity, thus reducing the resistance. The resistance range and the sensitivity of a photoresistor may vary considerably between different devices.



## Software

The lab kit software for the first part of this book is written mainly in the Python programming language. The available kit code has been tested and is compatible with Python version 2.7. For its operation, it requires the Grove Pi libraries<sup>3</sup>, which can be installed with a script provided by its manufacturing company through its repository on GitHub. In this repository, you can find complete instructions for installing it. The other required library is *python-forecastio*<sup>4</sup>.

The code for the lab activities follows a logic that is similar to Arduino code. The main function is called *main()*, which calls once the *setup()* function that is responsible for the initialization of each exercise, and the *loop()* function that undertakes the refresh and display of the data, repeating until the user stops running the program.

### • setup ()

*setup()* is responsible for initializing the exercise. Inside, the Grove Pi inputs and outputs (Buttons, Switch and Two-Color LEDs) are initialized, the LCD Monitor and the LEDs. Also in *setup()* is the connection with the sensor provider via the *init-Data()* function, as well as in some exercises, the first data reception by the provider through the *get [Sensor] Data()*

<sup>3</sup>

<sup>4</sup> <https://pypi.org/project/python-forecastio/>



function. Finally, there are some indicative information such as the User Name, the building and the rooms from which the measurements are, if they are environmental data.

- **loop ()**

The *loop()* function is responsible for updating and displaying the data, as well as managing the user's inputs from the Buttons and the Switch. In some exercises it also takes initial download of sensor data when it is not done in the *setup()* function using the *get [Sensor] Data()* function. The control of the input from the Buttons and the switch is done by the *checkButton()* and *checkSwitch()* respectively.

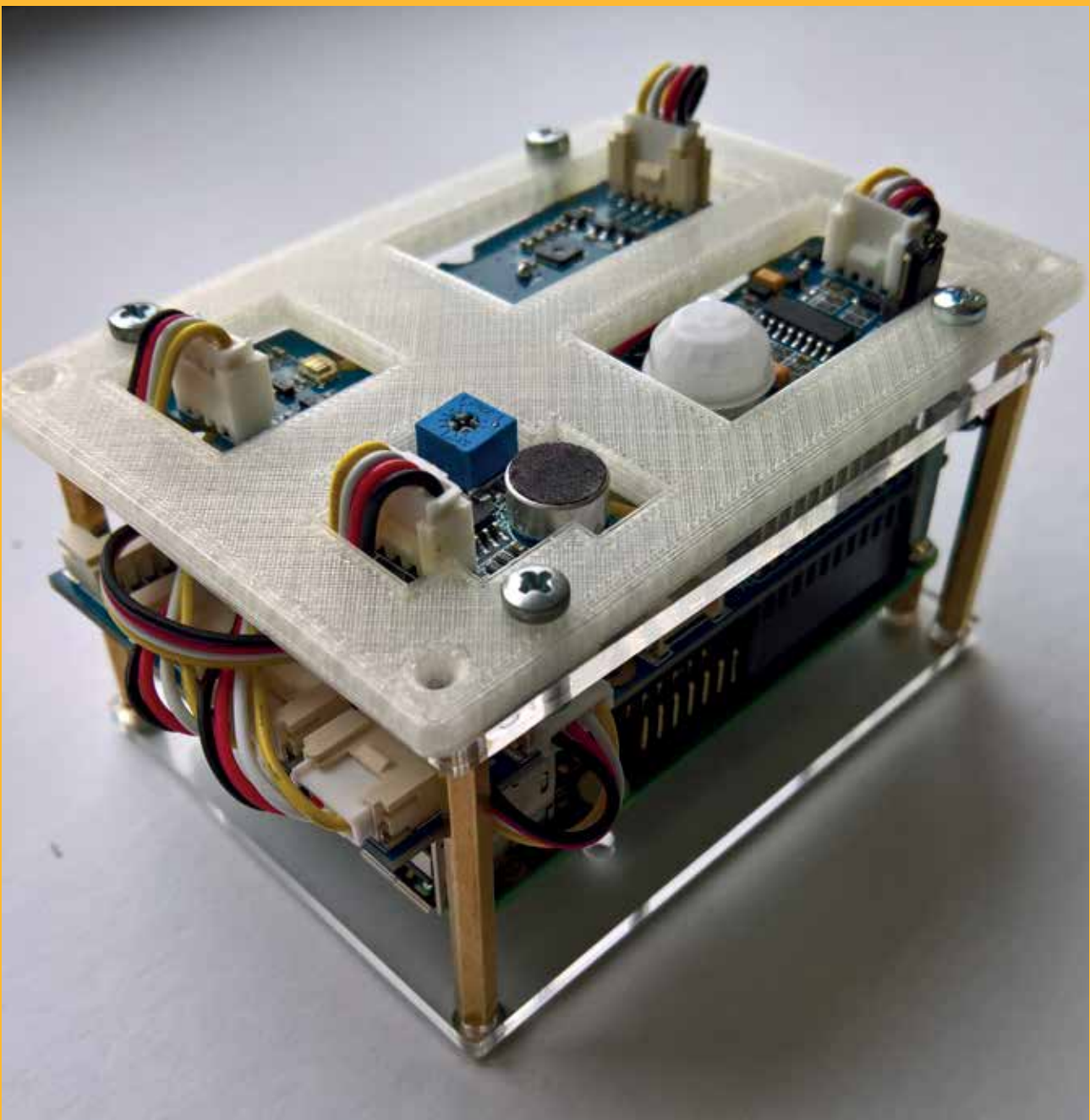
- **initData ()**

*InitData()* assumes connection to the sensor sensor provider. In the case of workshops, the provider is the SparkWorks

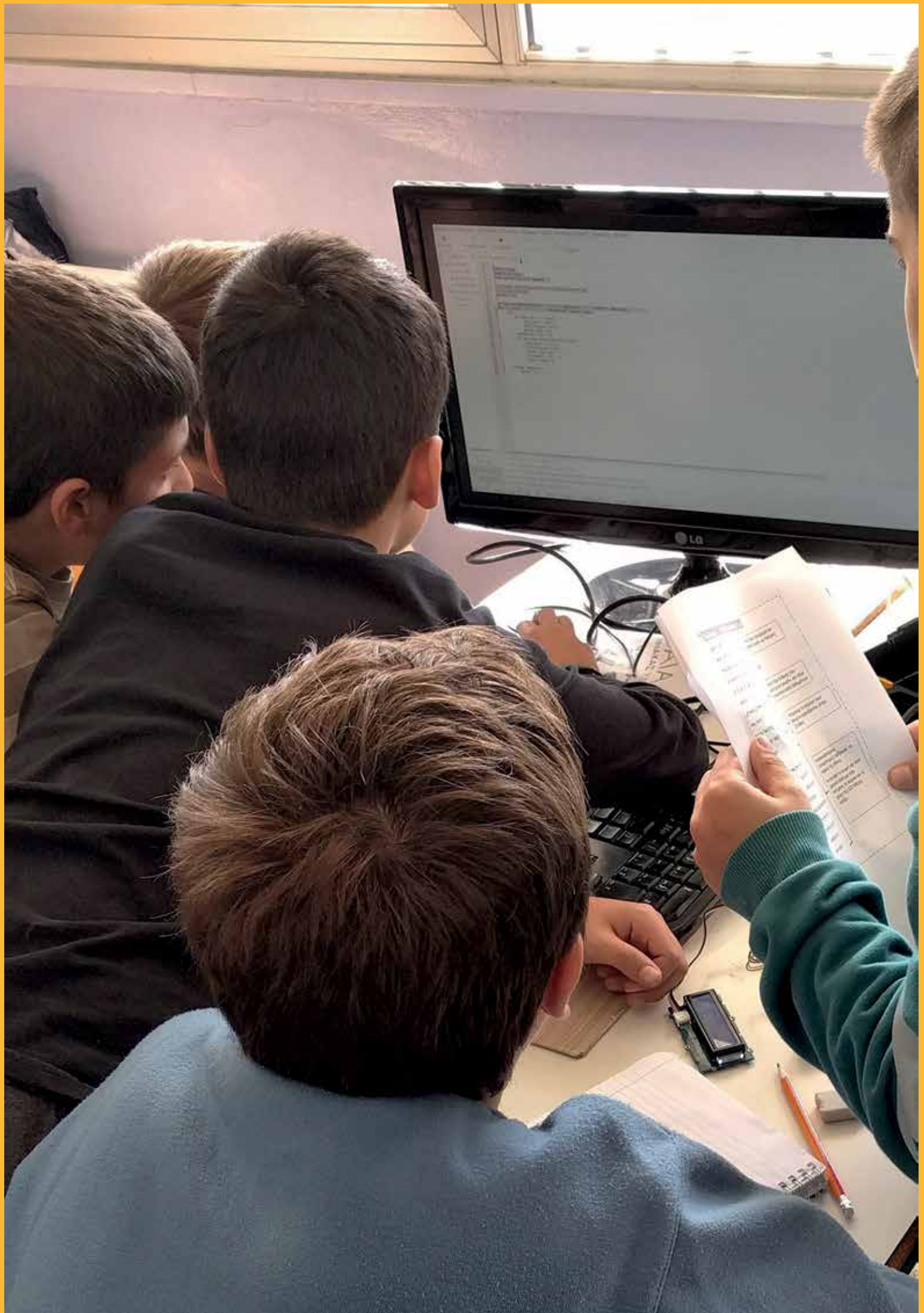
platform and communication is through the corresponding library in the exercise file. You can replace the application of this function to work with any provider you want. The function should return a variable with the building information as well as a list of variables with information of each room or phase depending on the exercise.

- **update [Site] Data ()**

*update [Site] Data()* takes over the data from the provider. You can replace its application to work with any provider you want. The *[Site] Data()* update in the application above SparkWorks requires the room as returned by the *initdata()* and a string with the name of the phenomenon for which metrics are requested. The function return a list with the last 48 values together with the latest value received for the specific sensor being monitored.









# CHAPTER 3: Preparatory Activities

## Computer Programming

### STUDENTS: Preparatory activity Level Beginner Computer Programming - Preparatory Activities

**1:** Using the keywords of the following table, search for Internet sources and collect information to answer the following questions:

- What is an algorithm?
- How can we command a computer to execute a program?
- What is the language of a computer?
- How can the computer understand our commands?

#### Main Concepts – Keywords:



Algorithm, Code, Compiler, Assembly, Command, Program

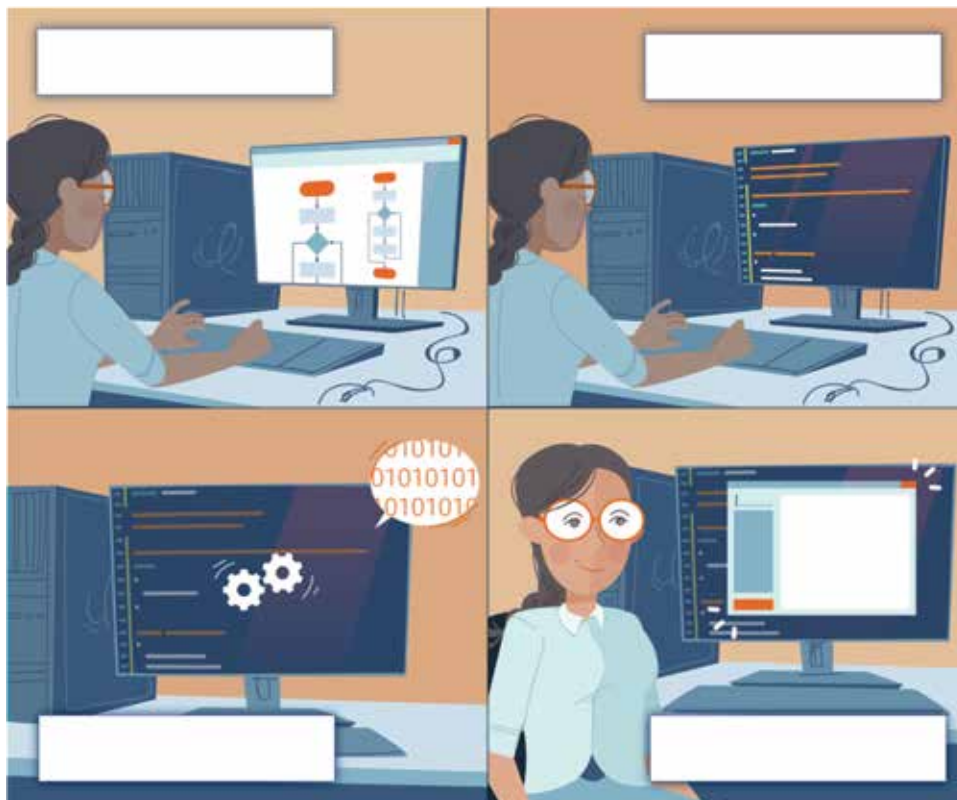
**2:** In the following pictures we see the stages a programmer follows to develop a computer program. Using the knowledge you're your research (1st question) and the concepts in the parenthesis, fill in the appropriate words in the bubble and the text boxes that follow.

#### Tools:

- Internet



(Compilation, Algorithm, Program Execution, Code Development)





## How is a computer program written?



! Links with information for programming and algorithms:

- <https://en.wikipedia.org/wiki/Algorithm>
- [https://en.wikipedia.org/wiki/Computer\\_programming](https://en.wikipedia.org/wiki/Computer_programming)
- [https://en.wikipedia.org/wiki/Programming\\_language](https://en.wikipedia.org/wiki/Programming_language)
- <https://en.wikipedia.org/wiki/Compiler>

### TEACHERS: Preparatory activity Level Beginner Computer Programming - Preparatory Activities

**1:** Using the keywords of the following table, search for Internet sources and collect information to answer the following questions:

- What is an algorithm?
- How can we command a computer to execute a program?
- What is the language of a computer?
- How can the computer understand our commands?

**2:** In the following pictures we see the stages a programmer follows to develop a computer program. Using the knowledge you're your research (1st question) and the concepts in the parenthesis, fill in the appropriate words in the bubble and the text boxes that follow.

(Compilation, Algorithm, Program Execution, Code Development)

#### Main Concepts – Keywords:



Algorithm, Code, Compiler, Assembly, Command, Program

#### Tools:



- Internet
- School Book



## How is a computer program written?



**An algorithm** is a series of commands that are well defined and can be used to solve a specific problem.

We execute such commands every day to prepare our meals, go to school, or decide what to do in our free time. In other words, an algorithm is the instructions used to execute a cooking recipe or to assemble a construction.

The commands of an **Algorithm** are instructions that are given to the computer through a written computer program. This procedure is called **programming**.

Programming takes into account also the testing of the program to verify its operation and its validity (debugging), the preparations of the instructions for the computer to execute the program and its requirements. There are thousands of **programming languages** through which people can communicate with the computer and synthesize a computer program.

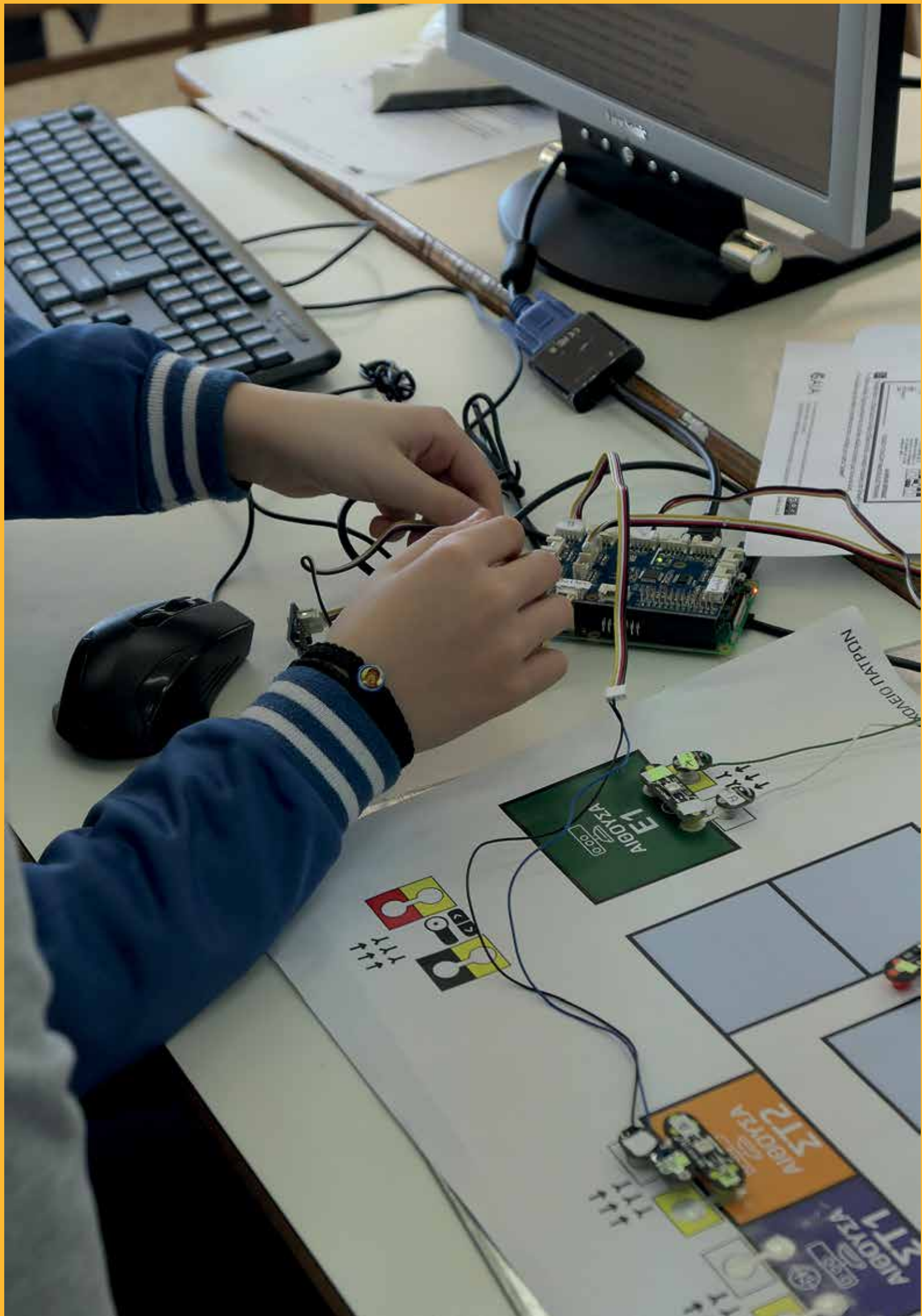
The programming languages look a lot like the language of humans. The commands included are mostly written in English, although there are some written in other languages

too, mainly for educational purposes. We can learn and understand programming languages to use them and control a computer as we desire. As far as the language that the computer speaks it is far away from what we know as a language, as it contains no letters. In fact, it is composed of only two numbers, zero (0) and one (1). But how can the computer understand us? The response to this question is given by a special program called **a compiler, or interpreter** that does exactly what its name means. It acts as an interpreter between humans and computers.

**Compiler or interpreter** is a computer program that reads a computer program written in a programming language (the source language that seems more like the language spoken by humans) and translates it to an equivalent code in another language, called machine language (assembly) or another programming language. The text input for the compiler is called a **source code** and the output is the **object code** known also as **machine language**.

In this way the computer can identify the commands we give during the execution of the program.







# CHAPTER 4: Introduction

## GrovePi+ Demonstration

### STUDENTS: 1st WorkSheet Level Beginner Introduction

- 1:** Connect the GrovePi+ to RaspberryPi (hook into the GPIO port) according to the following image (image 1).



Image 1: Grove Pi



Image 2: Rasberry Pi

- 2:** Connect the LCD screen and the button to the GrovePi+, according to the label “**Screen**” and “**Button**” respectively (see image 2), using the two cables.

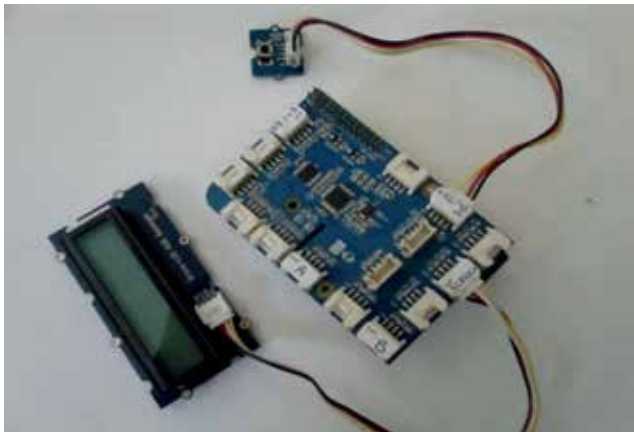


Image 3: Connection LCD monitor and Button

#### Main Concepts – Keywords:



Raspberry Pi, GrovePi+, Monitor LCD, Button, Loop, Automation.

#### Laboratory Equipment - Preparation


- Raspberry Pi
- GrovePi+
- Monitor LCD
- Button
- Adaptor HDMI to VGA
- Power Supply

- 3:** Connect the Raspberry Pi with the **keyboard**, the **mouse** (USB Port) and the **screen** using the Adaptor **HDMI to VGA**. Moreover, connect the power supply to the **micro USB port** and then **plug it into wall socket**. Wait for the interface to appear, then be sure that appears a confirmation message in the top right corner.



Image 4: Interface with the confirmation message to the right and the Geany icon to the left



As soon as the interface is ready, you can start the **Geany**  **Program**, by double-clicking the icon on the desktop.

**4:** In the Geany environment, choose the command Open on the toolbar and select the file bringing your school name, for example NeaPhiladelphia.

**5:** Double-click the document named GAIA-Workshop1.1.py check the code in the programming environment. In order to run the code, you have to select the “Run” button to the toolbar or press the F5 button on the keyboard.

**!:** Congratulations! If everything went well, you have run the program, now you can check the LCD monitor, the text changes color while pushing the button element.

**?:** Read the code on page 3 (Programming Code **GAIA-Workshop1.1.py**) and proceed with the text changing in order to appear in the desired text.

**?:** Let's Play with the colors! Find the right function in your code and change the colors that alternate on the LCD screen each time you press the button. You have to keep in mind that the final result is given by the value of red, green and blue (RGB – Red Green Blue).

**!:** Attention – The messages displayed on the LCD screen may only have Latin characters. Please ask your teacher what is the reason.

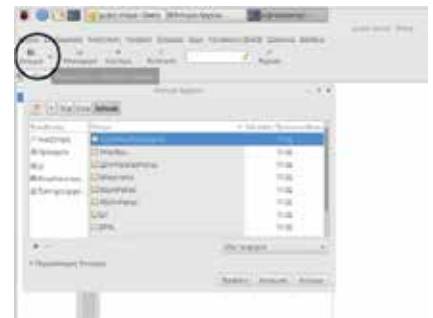


Image 5: The command “Open” to the Toolbar

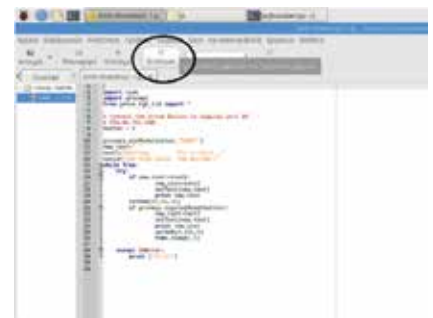


Image 6: The command “Run” in the Toolbar

Code – GAIA-Workshop1.1.py

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
import os
import sys
import time
sys.path.append(os.getcwd())
sys.dont_write_bytecode = True
import grovepi
import grove_rgb_lcd as grovelcd

# Connect the Grove Button
# to digital port D8
# SIG,NC,VCC,GND
button = 8

new_text = ""
text1 = "Waiting          for a click..."
text2 = "YOU HAVE CLICKED THE BUTTON!!!"

grovepi.pinMode(button, "INPUT")
grovelcd.setRGB(0, 0, 0)
grovelcd.setText("")

while True:
    try:
        if new_text is not text1:
            new_text = text1
            grovelcd.setRGB(50, 50, 50)
            grovelcd.setText(new_text)
            print(new_text)

            if grovepi.digitalRead(button):
                new_text = text2
                grovelcd.setRGB(0, 255, 0)
                grovelcd.setText(new_text)
                print(new_text)
                time.sleep(.5)

    except IOError:
        print("Error")
```

Import programs  
(GrovePi + Library & Python)

Defining the button connection to the  
GrovePi+.

Enter the text we want to  
appear in the LCD screen.

The button is entering data.  
Initialization of the LCD  
screen.

Repeatable (automation!)  
Define the text of the LCD  
screen.

While pressing the button  
and as long as keeping it  
pressed, the text and color  
of the LCD screen changes.



# GrovePi+ Presentation

## TEACHERS: 1st WorkSheet Level Beginner Introduction

### Suggested Answers

**?:** Read the code on page 3 (Programming Code GAIA-Workshop1.1.py) and proceed with the text changing in order to appear in the desired text.

### Answer

The variable `text2` (message) appears when you click the button element and the variable `text1` appears without clicking the button.

```
text2= "Your name"
```

**!:** Attention! Your text should always be in quotation marks “ ”

**?:** Let's Play with the colors! Find the right function in your code and change the colors that alternate on the LCD screen each time you press the button. You have to keep in mind that the final result is given by the value of red, green and blue (RGB – Red Green Blue).

The basic colors in the painting are red, yellow and blue. Correspondingly, in the computers the basic colors are red, green, blue. To achieve color mixing we use a different intensity of each color. To select the color on the LCD, you need to change the numbers in the following function:

```
grovelcd.setRGB(red, green, blue)
```

In each parameter we enter a level of basic colors we want to present using the numbers from 0 to 255. For instance, the red color is 255 to Red (r), 0 to green (g), and 0 to blue (b) position.

```
grovelcd.setRGB(255, 0, 0)
```

### Code – GAIA-Workshop1.1.py

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
import os
import sys
import time
sys.path.append(os.getcwd())
sys.dont_write_bytecode = True
import grovepi
import grove_rgb_lcd as grovelcd
```

```
# Connect the Grove Button
# to digital port D8
# SIG,NC,VCC,GND
button = 8
```

```
new_text = ""
text1 = "Waiting          for a click..."
text2 = "YOU HAVE CLICKED THE BUTTON!!"
```

```
grovepi.pinMode(button, "INPUT")
grovelcd.setRGB(0, 0, 0)
grovelcd.setText("")
```

```
while True:
```

```
    try:
```

```
        if new_text is not text1:
            new_text = text1
            grovelcd.setRGB(50, 50, 50)
            grovelcd.setText(new_text)
            print(new_text)
```

Select the color for the LCD screen when the button is not click

```
        if grovepi.digitalRead(button):
            new_text = text2
            grovelcd.setRGB(0, 255, 0)
            grovelcd.setText(new_text)
            print(new_text)
            time.sleep(.5)
```

Select the color for the LCD screen when the button is click

```
    except IOError:
        print("Error")
```



# Circuit Elements

## STUDENTS: 2nd WorkSheet Level Beginner Introduction

- 1:** Connect the GrovePi+ to RaspberryPi (hook into the GPIO port). Connect the Raspberry Pi with the keyboard, the mouse (USB port) and the screen (HDMI port) and turn it on by plugging it into the power cord.
- 2:** Use the conductive ink (pen) and draw the circuits on the schematic you were given (image 1).
- 3:** Place the schematic on the metal surface (image 1) and apply the LEDs to the appropriate positions, then the resistor and finally the "Switch" following the colors of the schematic. Using the two terminals and a resistor of 470K Ohm (yellow, purple, yellow), connect in the proper way the right part of the schematic.
- 4:** Place the magnetic connectors corresponding with colors, the magnetic connectors with the white color correspond to the white foursquare and the magnetic connectors with the yellow color correspond with the yellow foursquare. After that, place the "Switch" terminals in the right part of the schematic matching the colors at the top in the right way (the cables with three magnetic red, black, yellow).

### Main Concepts – Keywords:



Raspberry Pi, GrovePi+,  
electroninks LED, Switch,  
Resistor, Conductive ink,  
Initialize, Automation.

### Laboratory Equipment - Preparation



- Raspberry Pi
- GrovePi+
- Electroninks LED (2)
- Adaptor HDMI to VGA
- Power Supply
- Metal Surface
- Conductive ink
- Schematic
- Electroninks Switch
- Electroninks Connector with resistor
- Two terminal Cables (2)
- Three terminal Cables (1)

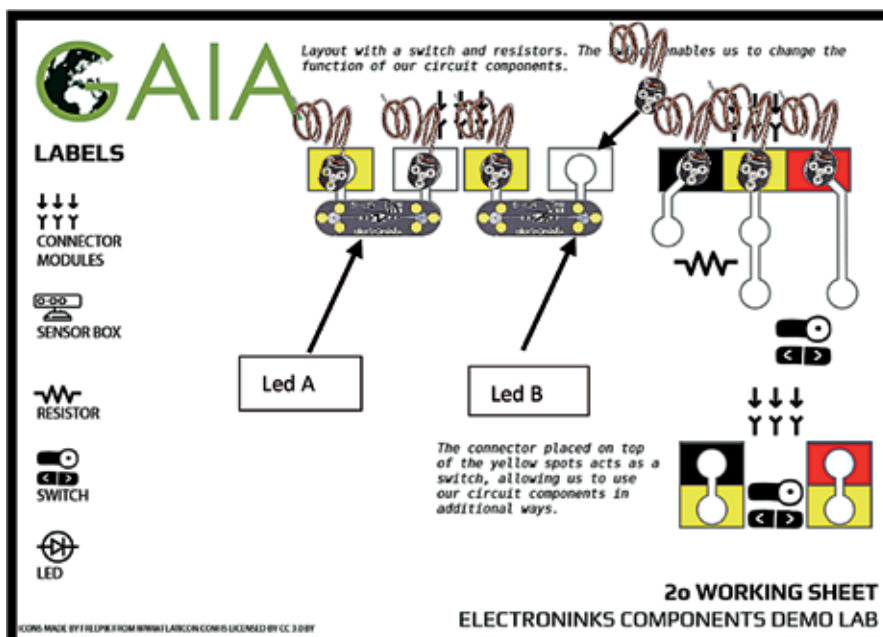


Image 1: Fitting Electroniks modules on the 1st worksheet circuit.



**5:** Place the fitting (electroninks connector modules) to the GrovePi+ (image 2) using the right connection cables (with the white terminals).

- Connect the screen cable to the label **“Screen”**
- The cable for the Switch operation is hooking into the port with label **“Switch”**
- The cable for the LED A is hooking into the port with label **A**
- The cable for the LED B is hooking into the port with label **B**


**6:** Start the **Geany**  **Program**, by double-clicking the icon on the desktop.



Image 2: The GrovePi+.



Image 3: Desktop displaying the confirmation message to the right top and the Geany icon to the left

**7:** In the **Geany** environment, choose the command **Open** on the toolbar and select the file bringing your school name, for example NeaPhiladelphia.

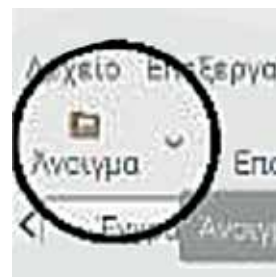


Image 4: Command Open on the toolbar

**8:** Double-click the document named **GAIA-Workshop1.1.py** check the code in the programming environment. In order to run the code, you have to select the **“Run”** button to the toolbar or press the F5 button on the keyboard.



Image 5: Command Run on the toolbar



?: Move the switch first to the left and then to the right. What's the result?

?: Try replacing the LED lights (i.e. the yellow terminal on the white connector and the white terminal on the yellow connector- see image 6). What do you notice?

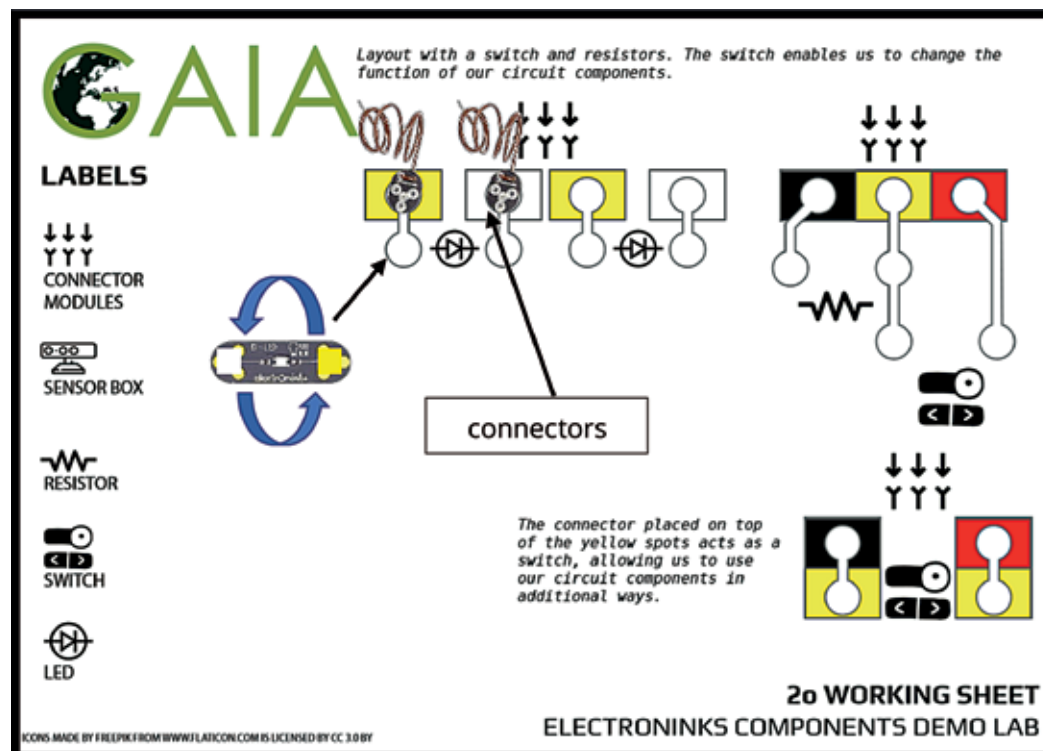


Image 6: Rotate the LED according to the example.

9: Check the code and comments below. Try changing the code to get the following:

?: During the previous steps we tried to change the LED's Colors placing it in the opposite way. Could you think a way to change the LED's color directly from the programming code?





## Code – GAIA-Workshop1.2.py

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
import os
import sys
import time
sys.path.append(os.getcwd())
sys.dont_write_bytecode = True
import grovepi
import grove_rgb_lcd as grovelcd
import gaia_text
```

Import Program (Library) for the use of circuit components.

```
switch = 0
# select pins for the leds
pins = {'A': {'yellow': 2, 'white': 3},
        'B': {'yellow': 4, 'white': 5},
        'C': {'yellow': 6, 'white': 7}}
```

Variable Initializations that define the ports using by the LEDs in the GrovePi+.

```
# select colors for the rooms
lcd_colors = [[255, 0, 255],
               [255, 128, 0],
               [0, 255, 0]]
```

```
text = ""
new_text = ""
exitapp = False
```

Assigned the port 0 as input to send the switch values to RPi and to change the program operation.

```
def setup():
    # Declare the Switch pins as INPUT
    grovepi.pinMode(switch, "INPUT")
    # Declare the LED pins as OUTPUT
    grovepi.pinMode(pins['A']['yellow'], "OUTPUT")
    grovepi.pinMode(pins['A']['white'], "OUTPUT")
    grovepi.pinMode(pins['B']['yellow'], "OUTPUT")
    grovepi.pinMode(pins['B']['white'], "OUTPUT")
    # Initialize the screen
    grovelcd.setRGB(50, 50, 50)
    grovelcd.setText("Starting..")

    time.sleep(1)
```

Assigned the port which contain the variable above to output ports (OUTPUT) in order to switch on the LED.

Initialize color and display text LCD.





```

def loop():
    global text, new_text
    value = grovepi.analogRead(switch)
    # interruptor off
    if value < 800:
        new_text = "All Leds Closed"
        # Close LED A
        grovepi.digitalWrite(pins['A']['yellow'], 0)
        grovepi.digitalWrite(pins['A']['white'], 0)
        # Close LED B
        grovepi.digitalWrite(pins['B']['yellow'], 0)
        grovepi.digitalWrite(pins['B']['white'], 0)
    # interruptor on
    else:
        new_text = "All Leds Open"
        # open led A color blue
        grovepi.digitalWrite(pins['A']['yellow'], 1)
        grovepi.digitalWrite(pins['A']['white'], 0)
        # open led B color blue
        grovepi.digitalWrite(pins['B']['yellow'], 1)
        grovepi.digitalWrite(pins['B']['white'], 0)

    if new_text is not text:
        text = new_text
        grovelcd.setText(text)
        print(text)

def main():
    setup()
    while not exitapp:
        loop()

try:
    main()
except KeyboardInterrupt:
    exitapp = True
    raise

```

Repeat function. When performing the routine, we check the position of the switch by reading its value. If it is in the Off position, the value read from port 0 is less than 800. Otherwise, it is in the On position the LEDs become blue.

Code that displays text on the LCD screen.

Main program function that performs initialization function setup () and repeat function loop () which displays text on the LCD screen.



## Suggested Answers

**?:** Move the switch to the left and then to the right. What's the result?

### Answer

*The LED lights turn off and on.*

**?:** Try replacing the LED lights (i.e. the yellow terminal on the white connector and the white terminal on the yellow connector- see image 6). What do you notice?

### Answer

*The normal LED light contains one diode that allows the current to cross through in only one direction. When that happens the light turns on.*

*Each diode is constitutively composed of a simple positive and negative charge contact. In one area there are many electrons – “negative charge” – and on the other there are many “holes”, as we call them in physics, that is, the absence of electrons.*

*If the current direction change, the light turns off. This happens when we apply negative voltage where due to the negative load, the current can't pass through it.*

*However, the LEDs we use, contain two diodes, which allows to take advantage of both directions of the current.*

**9:** Check the code and comments below. Try changing the code to get the following:

### Answer

Notes:

- The lines which begins with the symbol # are comments and don't run. Their role is to help us understand the code.

- Variable Clarification:

```
pins = {'A': {'yellow': 2, 'white': 3},
        'B': {'yellow': 4, 'white': 5},
        'C': {'yellow': 6, 'white': 7}}
```

*The variable “pins” are a Dictionary of Python. The dictionaries are analogues of lists with the added feature that their data can be called based on a unique user-defined name. The pins are mapped to three dictionaries, one for each port used. Each of these dictionaries contains the two terminals of each port.*

- The function `grovepi.digitalWrite(pin, value)` will write the value of the variable ‘value’ (1 or 0) at the output specified by the variable “pin”.
- To activate the LED, we need to give it a different voltage at each terminal.
  - o Activation in blue color (yellow pin is 1 and white pin is 0)
  - o Activation in red color (Yellow pin is 0 and white is 1)
  - o Deactivation (Yellow pin is 0 and white pin is 0)
- Opening the LED A in red

```
o grovepi.digitalWrite(pins['A']['yellow'],0)
o grovepi.digitalWrite(pins['A']['white'],1)
```





### Answer:

```
def loop():
    global text, new_text
    value = grovepi.analogRead(switch)
    # interruptor off
    if value < 800:
        new_text = "All Leds Closed"
        # Close LED A
        grovepi.digitalWrite(pins['A']['yellow'], 0)
        grovepi.digitalWrite(pins['A']['white'], 0)
        # Close LED B
        grovepi.digitalWrite(pins['B']['yellow'], 0)
        grovepi.digitalWrite(pins['B']['white'], 0)
    # interruptor on
    else:
        new_text = "All Leds Open"
        # open led A color blue
        grovepi.digitalWrite(pins['A']['yellow'], 1)
        grovepi.digitalWrite(pins['A']['white'], 0)
        # open led B color blue
        grovepi.digitalWrite(pins['B']['yellow'], 1)
        grovepi.digitalWrite(pins['B']['white'], 0)
```

Check the position of the switch.

This part of the code is executed when the switch is closed (OFF)

This part of the code is executed when the switch is open (ON)



# Control Panel

## STUDENTS: 3rd WorkSheet Level Beginner Introduction

- 1:** Connect the GrovePi+ to RaspberryPi (hook into the GPIO port). Connect the Raspberry Pi with the keyboard, the mouse (USB port) and the screen (HDMI port) and turn it on by plugging it into the power plug.
- 2:** Use the conductive ink (pen) and draw the circuits on the schematic you were given (image 1).
- 3:** Place the schematic on the metal surface (image 1) and apply the LEDs to the appropriate positions, then the resistor and finally the "Switch" following the colors of the schematic. Using the two terminals and a resistor of 470K Ohm (yellow, purple, yellow), connect in the proper way the right part of the schematic.

### Main Concepts - Keywords:



Raspberry Pi, GrovePi+, Electroninks LED, Switch, LCD Screen, Button, Resistor, Conductive ink

### Laboratory Equipment - Preparation



- Raspberry Pi
- GrovePi+
- LCD Screen
- Button
- Adaptor HDMI to VGA
- Power Supply
- Electroninks LED(3)
- Schematic
- Metal Surface
- Conductive ink
- Electroninks Switch
- Electroninks Resistor
- Two terminal Cables (3)
- Tree terminal Cables (1)

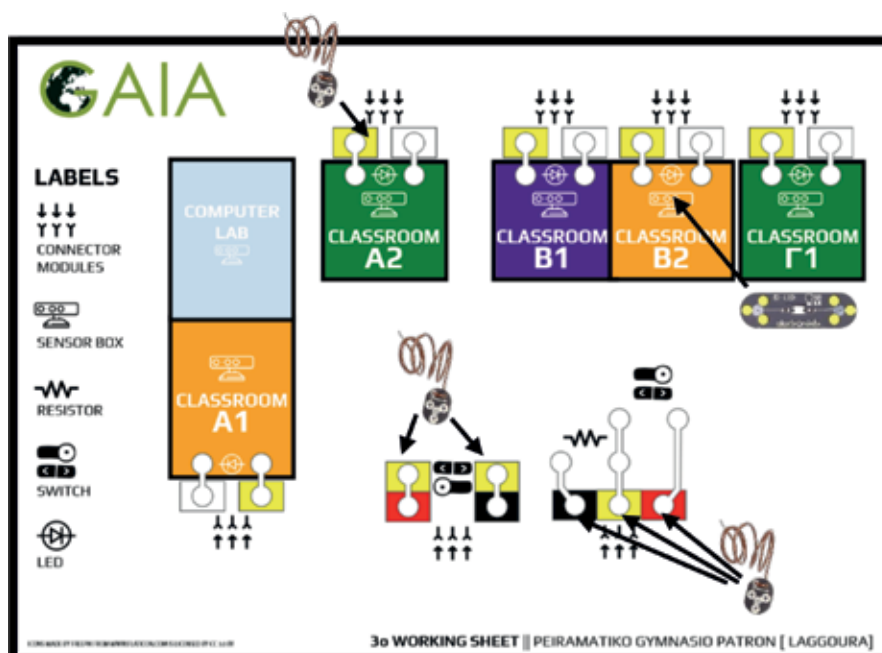



Image 1: Map 2nd floor Experimental Gymnasium School (Laggoura).



**4:** Place the electroniks connector modules to the GrovePi+ (image 2) using the right connection cables (with the white terminals).

- Connect the screen cable to the label “**Screen**”
- Connect the cable for the button into the port with label “**Button**”
- Connect the cable for the switch into the port with label “**Switch**”
- Connect the cable regarding the **purple** classroom into the port with label “**A**”
- Connect the cable regarding the **orange** classroom into the port with label “**B**”
- Connect the cable regarding the **green** classroom into the port with label “**C**”

**!:** Note that, just like in the second schematic, in the schools map has been given, there must be enough silver ink in some places. Let us remind you that its ink is conductive and plays the role of a cable on which all components must be placed accurately (components).

**5:** Start the **Geany**  **Program**, by double-clicking the icon on the desktop.

**6:** In the **Geany** environment, choose the command **Open** on the toolbar and select the file bringing your school name, for example NeaPhiladelphi.

**7:** Double-click the document named **GAIA-Workshop1.3.py** check the code in the programming environment. In order to run the code, you have to select the “**Run**” button to the toolbar or press the F5 button on the keyboard

**7:** Move the switch to the left and right. What's the result?

**?:** How could you use the sensors measurement from the classrooms in order to reduce your schools energy consumption? Put your first ideas on and distribute your suggestion with your school team. Be sure there will be useful for the next workshops!

**?:** Do you want to change the upper or base limit, which sensors value should have? Find the code in the Teacher Worksheet and change the limits in the if conditional.



Image 2: The GrovePi+.



Image 3: Desktop displaying the confirmation message to the right top and the Geany icon to the left

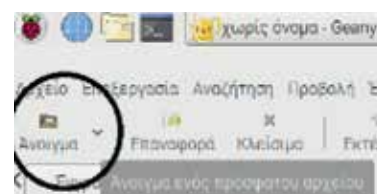


Image 4: Command Open on the toolbar

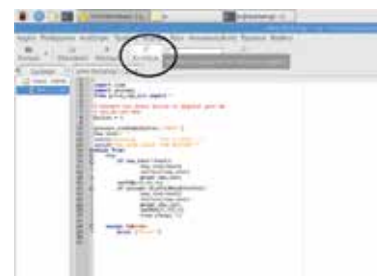


Image 5: Command Run on the toolbar



**Suggested Answers**

**8:** Move the switch to the left and right. What's the result?

**Answer**

**Switch Position left:** Every time you press the button, we choose different sensor from the classrooms modifying the values on the LCD screen and moreover the role of the LED lights from the map. For every sensor, appears the value of the measurement in the room with the corresponding color on the LCD screen (for instance: purple for the classroom A). The LED lights turns **red** when the value overcomes the normal defined value (the up and down limit) through the code execution. For example, if the temperature is over 25 or above 18 degrees the red light turns on. If the values recorded by the sensor are normal, the color of the LED becomes **blue**.

**Switch Position right:** Every time you press the button, appears in the LCD screen the maximum and minimum value for every sensor placed on the classroom.

**Note that the value from the three classrooms are shown as a table on the Raspberry Pi console.** The classroom with the maximum or minimum value for the selected sensor is **red**.

**?:** Do you want to change the upper or base limit, which sensors value should have? Find the code below (page 4) and change the limit of the if conditional.

**Answer**

```
def showLuminosity(light_value, a, b):
    if (light_value < 200):
        # red LED
        grovepi.digitalWrite(a, 0)
        grovepi.digitalWrite(b, 1)
    else:
        # blue LED
        grovepi.digitalWrite(a, 1)
        grovepi.digitalWrite(b, 0)
```

Control function of the classrooms luminosity. From the command if seems that control condition «turns red» when the light value is lowest than 200.

```
def showTemperature(temperature_value, a, b):
    if 18 < temperature_value < 25:
        # Blue LED
        grovepi.digitalWrite(a, 1)
        grovepi.digitalWrite(b, 0)
    else:
        # red led
        grovepi.digitalWrite(a, 0)
        grovepi.digitalWrite(b, 1)
```

Control function of the classrooms' temperature. From the command if seems that control condition «turns blue» for the values light from 18°C until

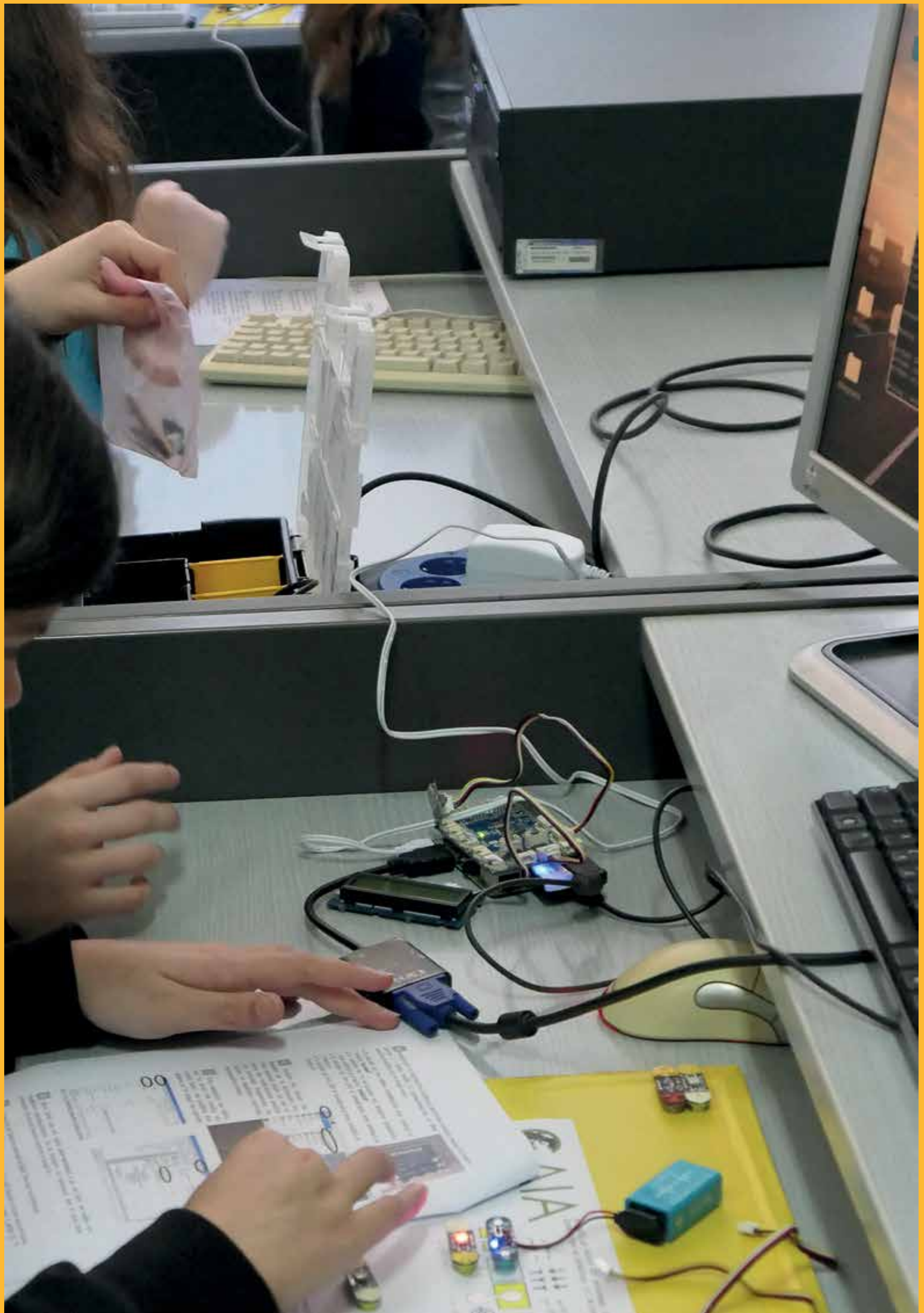
```
def showHumidity(humidity_value, a, b):
    if 30 < humidity_value < 50:
        grovepi.digitalWrite(a, 1)
        grovepi.digitalWrite(b, 0)
    else:
        grovepi.digitalWrite(a, 0)
        grovepi.digitalWrite(b, 1)
```

Control function of the classrooms' humidity. From the command if seems that control condition becomes «blue» the light for values from 31% to 49%.

```
def showNoise(noise_value, a, b):
    if noise_value < 50:
        grovepi.digitalWrite(a, 1)
        grovepi.digitalWrite(b, 0)
    else:
        grovepi.digitalWrite(a, 0)
        grovepi.digitalWrite(b, 1)
```

Control function of the classrooms' noise. From the command if seems that control condition «turns red» the light for values from 50 and above.







# CHAPTER 5: Lighting

## Light Sensors

### STUDENTS: 1st WorkSheet Level Beginner Lighting

**1:** Connect the GrovePi+ to RaspberryPi (hook into the GPIO port). Connect the Raspberry Pi with the keyboard, the mouse (USB port) and the screen (HDMI adaptor) and turn it on by plugging it into the power plug.

**2:** Using the conductive ink, draw the circuits on the map you were given.

**3:** Place the map on the metal surface and the connector modules, for example the **Electroninks connectors** with resistor 10K (brown, black, red) and the **Electroninks connectors** with photoresistor at the appropriate map position (Figure 1). Remember who you put the connection cables in the previous labs (black, red and yellow magnet positions).

**!:** Attention! First the magnets and then the clips.

**4:** When placing the magnetic parts of the cable connections (connectors) on the map, "connect" their clips into the appropriate GrovePi+ positions by matching them with the appropriate colours (Figure 2).

### Main Concepts – Keywords:



Raspberry Pi, GrovePi+, Electroninks LED, Photoresistor, Conductive ink, Light sensor (LED), Lux

### Laboratory



### Equipment - Preparation

- Raspberry Pi
- GrovePi+
- Adaptor HDMI to VGA
- Power Supply
- Schematic
- Metal Surface
- Conductive ink
- Electroninks Connector with photoresistor
- Electroninks Connector with resistor 10K
- Light sensor (photodiode)
- Light sensor (photo resistor)

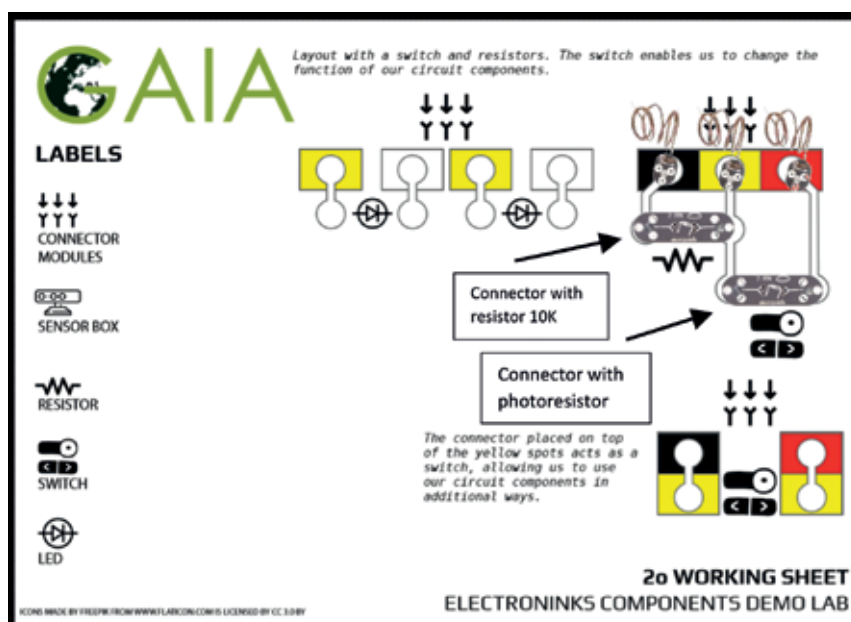


Figure 1: Fitting Electroninks modules on the 1st worksheet circuit.



**5:** Install the connector modules (Electroninks connector modules) to Grove-Pi+ (figure 2) using the special connecting cables (with white clip).

- The cable for the light sensor is placed in the slot labeled **“Lux”**
- The cable for the Connector with the photoresistor is placed in the slot labeled **“D~”**.

**6:** As soon as the interface is ready, start the **Geany** 🍷 **Program**, by double-clicking the icon on the desktop.

**7:** In the Geany environment, choose the command **Open** on the toolbar and select the folder with your school's name (e.g. **NeaPhiladelphia**).

**8:** Double-click the **GAIA-Workshop2.1.py** file and see the code in the programming environment. In order to run the code you have to select the **Run** button to the toolbar or press the F5 button on the keyboard.

**!:** By running the program you can see the brightness of the point at which you are located, in Lux units from the photodiode but also from the photoresistor in approach.

In this way, you can match the levels that the photoresistor perceives by measuring in Lux units with the Photodiode.

**?:** Use your own light sources to see the sensor readings. You will be impressed! Compare the values returned by the photoresistor and those of the light sensor in terms of accuracy and range (maximum – minimum value). Where do they differ;

Photoresistor	Photodiode (Lux)
0	
12	
33	
75	
200	

**?:** Mark the indications you get from the lighting of the room you are in with lights on and off. Keep these values and compare them with the one you will get from the photoresistor of the IoT device were installed in your class (2nd worksheet).

	Photoresistor	Photodiode
Lights On		
Lights Off		

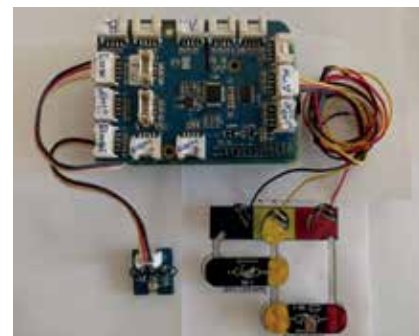


Figure 2: The GrovePi+ with the modules.



# Demonstration of GrovePi+

## TEACHERS: 1st WorkSheet Level Beginner Lighting

This worksheet does not value in schools where their devices use photodiode because the measurements are already in Lux.

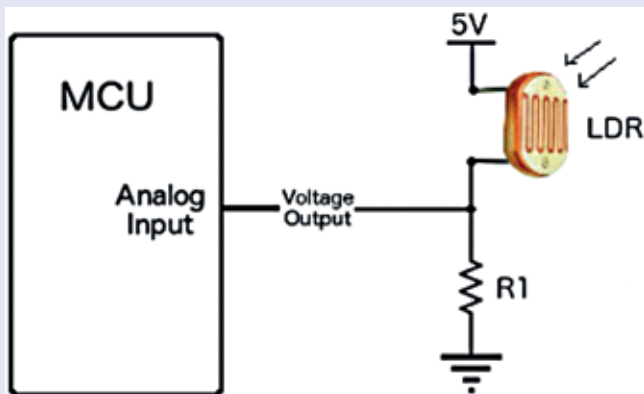
### Suggested Answers

! : Some general Knowledge – What are photoresistors?

#### Answer

Photoresistors, also known as light dependent resistors (LDR), are light sensitive devices most often used to indicate the presence or absence of light or to measure the light intensity. In the dark, their resistance is very high, sometimes up to  $1M\Omega$ , but when the LDR sensor is exposed to light, the resistance drops dramatically, even down to a few ohms (depending on the light intensity).

Circuit: We use a  $R1=10K\Omega$ . We connect the Raspberry Pi to A1 to read the voltage changes corresponding to the changes in the resistance of the LRD.



What is the photodiode?

#### Answer

The photodiodes typically perform the opposite effect on LEDs and laser diodes. Instead of using electricity to cause the combination of electrons and holes to form photons, the photodiodes absorb light energy (photons) to create pairs of electrons-holes, thereby creating an electric current flow. They have more sensitivity than photo resistances and allow us to give prices to LUX.

What is Lux?

#### Answer

The lux (symbol: lx) is the commonly-used unit of illuminance and luminous emittance, measuring luminous flux per unit area. It is equal to one lumen per square metre. In photometry, this is used as a measure of the intensity, as perceived by the human eye, of light that hits or passes through a surface.

? : Use your own light sources to see the sensor readings. You will be impressed! Compare the values returned by the photoresistor and those of the light sensor in terms of accuracy and range (maximum-minimum value). Where do they differ?

#### Answer

The corresponding values between photoresistor and photodiode are pretty much the following. This table will help you calculate the lighting in your room in Lux units:

Photoresistor	Photodiode
0	0-30 Lux
12	30-70 Lux
33	70-130 Lux
75	130-300 Lux
200	> 300Lux



# Light control

## STUDENTS: 2nd WorkSheet Level Beginner Lighting

- 1: Using the conductive ink draw the circuits on the map you were given.
- 2: Place the map on the metal surface.
- 3: Find 3 LED from the workshop kit and place them in each room matching the colors.
- 4: Find the Raspberry Pi from the workshop kit, and then connect the wires. We will use 3 sets of wires with 2 magnetic connectors, matching the colors yellow and white to the squares of the map.
- 5: Connect the cables according to the diagram below:

### Main Concepts – Keywords:



Raspberry Pi, GrovePi+, Electroninks LED, Screen LCD, Button, Conductive ink, Lighting

### Laboratory Equipment - Preparation



- Raspberry Pi
- GrovePi+
- 1 Screen LCD
- 1 Button
- Adaptor HDMI to VGA
- Power Supply
- Two terminal cables (3)
- Electroninks LED (3)
- Schematic
- Metal surface
- Conductive ink

**Purple** Classroom → Label **A**

**Orange** Classroom → Label **B**

**Green** Classroom → Label **C**

- 6: Then connect:
  - The LCD screen cable in the port labeled “screen”
  - The button cable in the port labeled “button”

- 7: Open the **Geany**  **Environment**

- 8: In the Geany environment, select the **Open** command on the toolbar, and go to the folder with your school name.

- 9: Double click in the **GAIA-Workshop2.2.py** file and see the code in the programming environment. To run the program, select the **Run** command on the toolbar or press the F5 button on the keyboard.

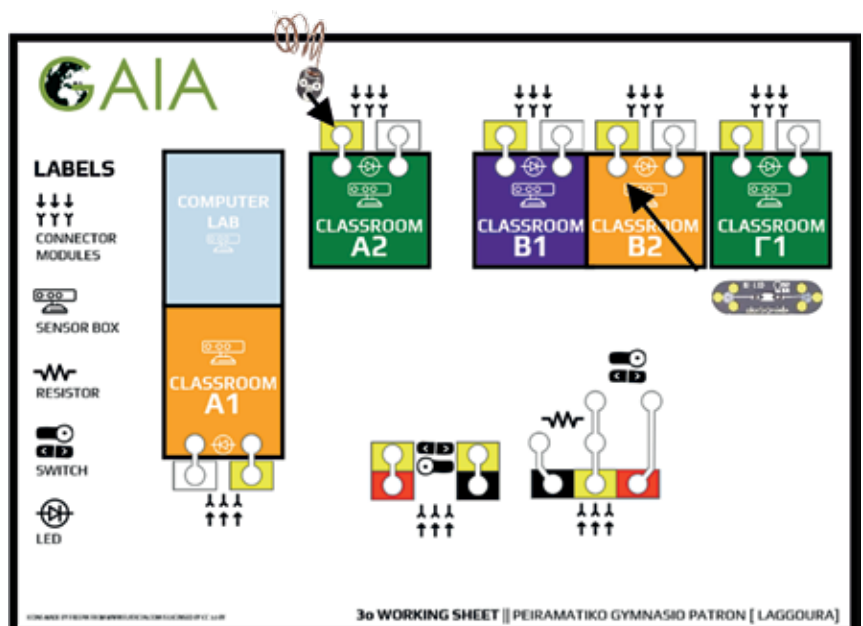


Figure 1: Floor Plan of 2nd Floor of the Experimental High School of Patras (Laggoura).



**?:** Observe the measurements displayed on the LCD screen every time you press the button. Can you describe them?

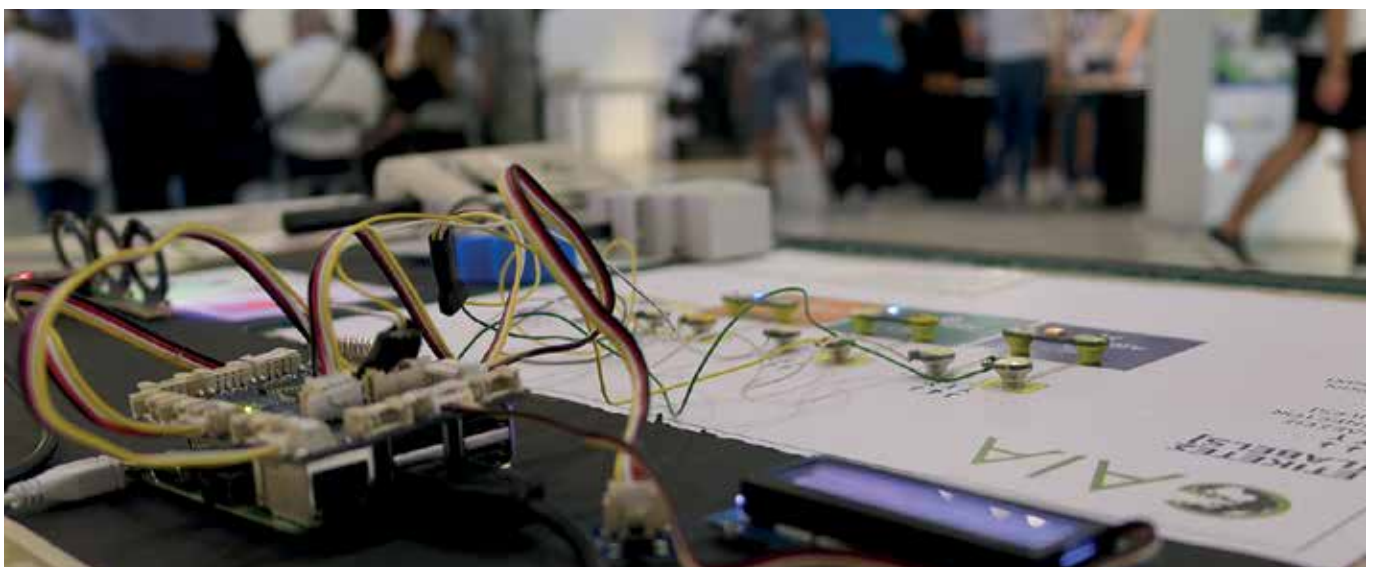
Classroom <b>Purple</b> →	
Classroom <b>Orange</b> →	
Classroom <b>Green</b> →	
Maximum lighting	
Maximum lighting	

**?:** Turn on the lights and write down the new values in the table. Turn off the lights and write down the new values in the table.

	Lights On	Lights Off
Classroom <b>Purple</b> →		
Classroom <b>Orange</b> →		
Classroom <b>Green</b> →		
Maximum lighting		
Maximum lighting		

- Which classroom has the most light with turned lights off? Why?  
.....
- Which classroom has the most light with turned lights on? Why?  
.....
- In your opinion, which classroom has the most artificial light and which one has the most natural light?  
.....

**?:** Which classrooms are adequately illuminated by natural light knowing that the adequate level of lighting is greater than 200 Lux?;





**Suggested Answers**

**?:** Observe the measurements displayed on the LCD screen every time you press the button. Can you describe them?

**Answer**

With each click of the button, we switch to the next mode. The functions are described below:

**Function 1:**

Shows the light intensity in the Purple classroom. The corresponding LED is **red** if the intensity of light value is less than 200 (the brightness of the classroom is not sufficient) and **blue** if the brightness of the classroom is sufficient (200 is the value of adequate lighting in a classroom).

**Function 2:**

Shows the light intensity in the Orange classroom. The corresponding LED is **red** if the intensity of light value is less than 200 (the brightness of the classroom is not sufficient) and **blue** if the brightness of the classroom is sufficient (200 is the value of adequate lighting in a classroom).

**Function 3:**

Shows the light intensity in the Green classroom. The corresponding LED is **red** if the intensity of light value is less than 200 (the brightness of the classroom is not sufficient) and **blue** if the brightness of the classroom is sufficient (200 is the value of adequate lighting in a classroom).

**Function 4:**

It shows which classroom between the 3 has the greatest

brightness by lighting up the corresponding LED in **blue** color and showing that value on the LCD screen.

**Function 5:**

It shows which classroom among the 3 has the least brightness by lighting the corresponding LED in **red** color and showing that value on the LCD screen.

**?:** Turn on the lights and write down the new values in the table. Turn off the lights and write down the new values in the table.

**Answer**

Talk with the students about the factors that can affect the lighting of each classroom. Such factors may be the orientation of each classroom, the floor where it is located, as well as the layout of the room, the curtains or furniture in the classroom.

**?:** Which classrooms are adequately lighting by natural light knowing that the adequate level of light intensity is greater than 200 Lux?

**Answer**

Continue the discussion from the previous question declaring what are the hallmarks of the classrooms that have adequate lighting as well as what can be improved in those that have not.





# Record history

## STUDENTS: 3rd WorkSheet Level Beginner Lighting

**1:** Using the table and the components of the 2nd worksheet, place a second button to the “**Button 2**” clip besides the “**Button**”.

**2:** Now we will use only 2 classrooms and 2 LEDs connected to port **A** and **B**. Remove the cable from port **C**.

**3:** Open the **GAIA-Workshop2.3.a.py** file in your school folder and see the code in the programming environment.

**4:** To run the program, select the **Run** icon on the toolbar or press the F5 button.

**!:** Pressing the “**Button**” you moved to the previous **five minutes**. When you press the “**Button 2**” you are changing a classroom.

**!:** The measurements you can see are only for the first two classrooms and not for the third.

**5:** Using the program measurements you just performed, note the luminosity of the classrooms during the first in the 1st table of brightness recording.

**1st TABLE OF LUMINOSITY RECORDING – 1st BIG BREAK**

Hour	Classroom .....	Classroom .....	Classroom .....	Classroom .....	Classroom .....

**?:** Is there a difference in lighting during the break and during the lesson? In your opinion during breaks, what should be done to reduce consumption?

.....  
.....

**6:** In the same way you worked before, open the **GAIA-Workshop2.3.b.py** file in your school folder and see the code in the programming environment. To run the program, select the “**Run**” icon on the toolbar or press the F5 button.

### Main Concepts – Keywords:



Raspberry Pi, GrovePi+,  
Electroninks LED, Screen LCD,  
Button, Conductive ink.

### Laboratory Equipment - Preparation



- Raspberry Pi
- GrovePi+
- Screen LCD
- Button (2)
- Electroninks LED (2)
- Schematic
- Metal interface
- Conductive ink
- Two terminal cables (2)



**!:** Pressing the “**Button**” you moved to the previous **hour**. When you press the “**Button 2**” you are changing a classroom.

**7:** Using the program measurements you just performed, note in the 2nd table of brightness recording, the maximum and minimum brightness in the classrooms for the day before. You can measure the natural lighting at times when the school does not work, as the electric lights should be closed. Then circle the maximum value of all you have noted.

2nd TABLE OF BRIGHTNESS RECORDING – NATURAL LIGHT					
Hour	Classroom .....	Classroom .....	Classroom .....	Classroom .....	Classroom .....
06:..					
10:..					
12:..					
14:..					
16:..					
18:..					
20:..					

**?:** Can you justify the high or low luminosity in these classrooms?

.....

.....

**?:** Taking into account the geographic position and orientation of your school, can you explain the changes in luminosity in each classroom during the day?

.....

.....

**?:** Based on what we saw in the 2nd exercise on the level of luminosity that the classrooms should have, circle the classrooms that had a brightness of 200. What hours is the use of lights necessary? Record the hours for each classroom.

Classroom .....:

.....

Classroom .....:

.....



### Suggested Answers

!:

 The measurements you can see are only for the first two classrooms and not for the third.

### Note

When the **"Button 2"** is connected, you can not associate room **"C"** at the same time. The reason for this is because these two ports share common inputs / outputs in the **Grove Pi** circuit.

5:

 Using the program measurements you just performed, note the luminosity of the classrooms during the first in the 1st table of luminosity recording.

### Answer

On the four lines of the panel, students should record the luminosity of the classrooms shortly before, during and shortly after the break. In this way, they can see if they remembered to close the lights or not.

?:

 Is there a difference in lighting during the break and during the lesson? In your opinion during breaks, what should be done to reduce consumption?

### Answer

Observe with students the lighting of the classroom during the break. If intensity of light remains high during the break, check whether this is due to physical factors or the lights remains on. Discuss with the students if it is necessary to have the lights on in the classroom or the natural light is enough.

7:

 Using the program measurements you just performed, note in the 2nd table of brightness recording,

the maximum and minimum intensity of light in the classrooms for the day before. You can measure the natural lighting at times when the school does not work, as the electric lights should be closed. Then circle the maximum value of all you have noted.

?:

 Can you justify the high or low brightness in these classrooms?

### Answer

Observe with students the school opening hours as well as the use of these classrooms.

?:

 Taking into account the geographic position and orientation of your school, can you explain the changes in brightness in each classroom during the day?

### Answer

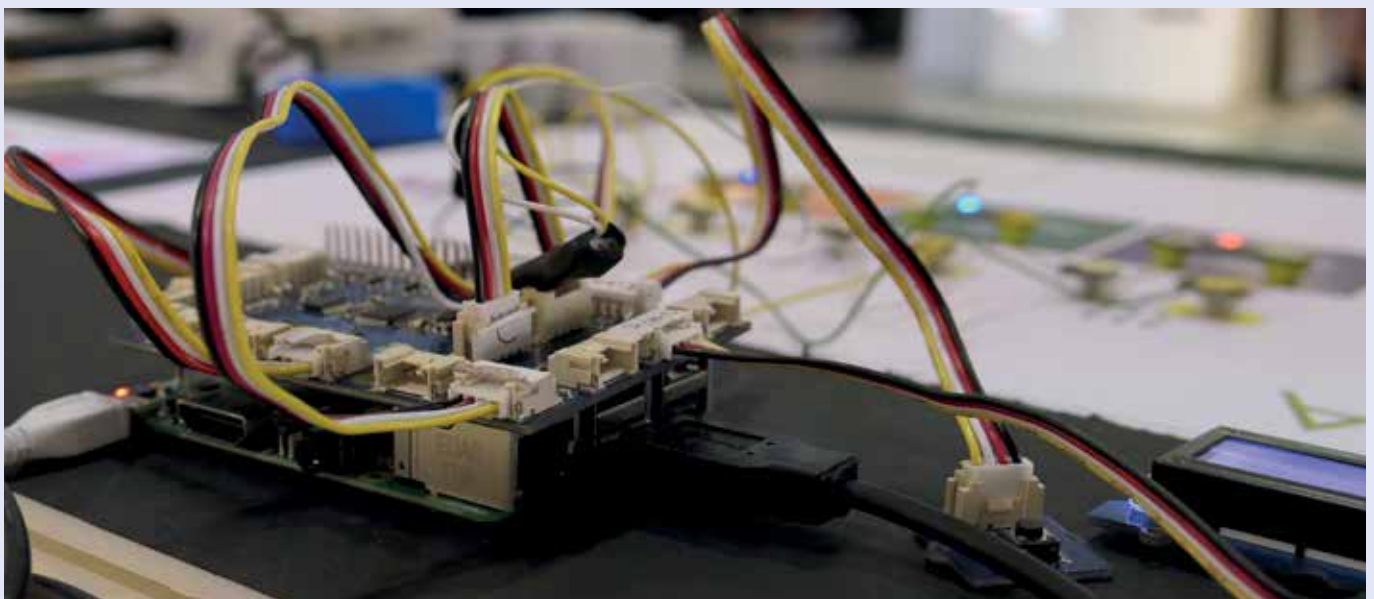
The purpose of this question is to show the importance of location, orientation and architectural / urban design of the school building. Because the schools are operating in the early hours, the eastern classes have more light. Other factors that can affect include the size and layout of the windows, adjoining buildings, geographic features such as the slope of a mountain.

?:

 Based on what we saw in the 2nd exercise on the level of brightness that the classrooms should have, circle the classrooms that had a brightness of 200. What hours is the use of lights necessary? Record the hours for each classroom in brightness in each classroom during the day?

### Answer

We consider adequate lighting to be from 200 and above.







Εκπομπές Πραγματικής Κατανάλωσης

Συνολική Κατανάλωση 1

46  $\mu\text{Wh/m}^2$

711  $\text{t CO}_2$

Κατανάλωση Ενέργειας, μέση

36  $\text{kWh/m}^2$

Κατανάλωση Ενέργειας, ανά κτίριο

29  $\text{kWh/m}^2$

Κατανάλωση Ενέργειας, ανά τ.μ.

0  $\text{kWh/m}^2$

Προβλεπόμενη Κατανάλωση Ενέργειας

Συνολική Κατανάλωση 2

274  $\mu\text{Wh/m}^2$

1185  $\text{t CO}_2$

Κατανάλωση Ενέργειας, μέση

880  $\text{kWh/m}^2$

Κατανάλωση Ενέργειας, ανά κτίριο

3.21  $\text{kWh/m}^2$

Κατανάλωση Ενέργειας, ανά τ.μ.

0.74  $\text{kWh/m}^2$



10



## CHAPTER 6: Temperature

# Monitoring Temperature - Humidity

### STUDENTS: 1st WorkSheet Level Beginner Temperature

**1:** Connect the GrovePi+ to RaspberryPi (hook into the GPIO port). Connect the Raspberry Pi with the keyboard, the mouse (USB port) and the screen (HDMI port) and turn it on by plugging it into the power plug.

**2:** Use the conductive ink (pencil) and draw the circuits on the map you were given (image 1).

**3:** Place the map on a metal surface.

**4:** Find 3 LEDs from the workshop kit and place them in each room matching the colors.

**5:** Find the Raspberry Pi from the workshop kit, then connect the wires, we will use 3 sets of wires with 2 magnetic connectors, by matching the colors yellow and white to the squares of the map.

**6:** Connect the cables according to the diagram below:

**Purple** Classroom → Label **A**

**Orange** Classroom → Label **B**

**Green** Classroom → Label **C**

**7:** Then connect:

- The LCD screen cable in the port labeled “screen”
- The button cable in the port labeled “button”

**8:** start the **Geany**  **Environment**

**9:** In the **Geany** environment, select the Open command on the toolbar, and go to the folder with your school name.

### Main Concepts – Keywords:



Raspberry Pi, GrovePi+,  
Electroninks LED, LCD screen,  
Button, Temperature Sensor,  
Humidity Sensor

### Laboratory Equipment - Preparation



- Raspberry Pi
- GrovePi+
- LCD Screen
- Button
- Adaptor HDMI to VGA
- Power Supply
- Electroninks LED (3)
- Schematic
- Metal Surface
- Conductive ink
- Two terminal Cables (3)

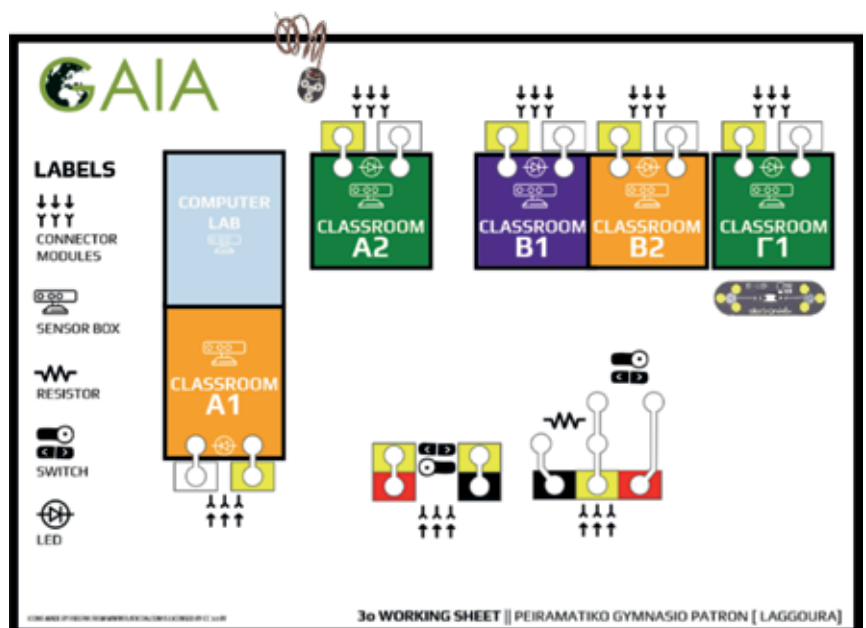


Figure 1: Floor plan of 2nd Floor of the Experimental High School of Patras (Laggoura)



**10:** Double-click the **GAIA-Workshop3.1.py** file and see the code in the programming environment. To run the program, select the **Run** command on the toolbar or press the F5 button on the keyboard.

**!:** The measurements you get come from the temperature and humidity sensors that are placed in the respective classrooms at your school.

**?:** Observe the measurements displayed on the LCD every time you press the button. In the 1st table, record the classroom with the minimum temperature and maximum humidity as well as their values. Can you describe them?

1st Monitoring Table Temperature – Humidity		
	Classroom	Valua
Minimum temperature (°C)		
Maximum humidity (°C)		

**?:** What is happening at this moment in your classrooms? Discuss it with your team.

.....  
.....

## TEACHERS: 1st WorkSheet Level Beginner Temperature

### Suggested Answers

**?:** Observe the measurements displayed on the LCD every time you press the button. In the 1st table, record the classroom with the minimum temperature and maximum humidity as well as their values. Can you describe them?

### Answer

When the **GAIA-Workshop3.1.py** program runs, the LCD screen shows the minimum temperature and maximum humidity values as following. Firstly, shows the minimum temperature between the three classrooms and the temperature of each classroom individually. Then the maximum humidity is displayed in the same was as above. While the LCD screen shows the temperature measures, LEDs on the map represent the room with the minimum temperature in **red** and the others in blue. Respectively for humidity is represented with LED in **red** at the classroom with the maximum humidity.

**?:** What is happening at this moment in your classrooms? Discuss it with your team.


### Answer

The temperature and humidity of a classroom are possible affected by its orientation, its location within the building, the environmental conditions and the way it is conditioned.



# Heat Index & Discomfort Index

## STUDENTS: 2nd WorkSheet Level Beginner Temperature

**1:** Using the map and the components of the 1o worksheet and always through the **Geany**  **environment**, open the **GAIA-Workshop3.2.a.py** file in your school folder and view the code in the programming environment.

**!:** The program calculates the "heat index" in the classrooms at this time.

### Main Concepts – Keywords:



Raspberry Pi, GrovePi+, Electroninks LED, LCD screen, button, LED rings, Temperature Sensor, Humidity Sensor, Discomfort index, Heat Index

### Laboratory Equipment - Preparation



- Raspberry Pi
- GrovePi+
- LCD Screen
- Button
- Adaptor HDMI to VGA
- Power Supply
- Electroninks LED (3)
- Schematic
- Metal Surface
- Conductive ink
- Two terminal Cables (3)
- LED Rings

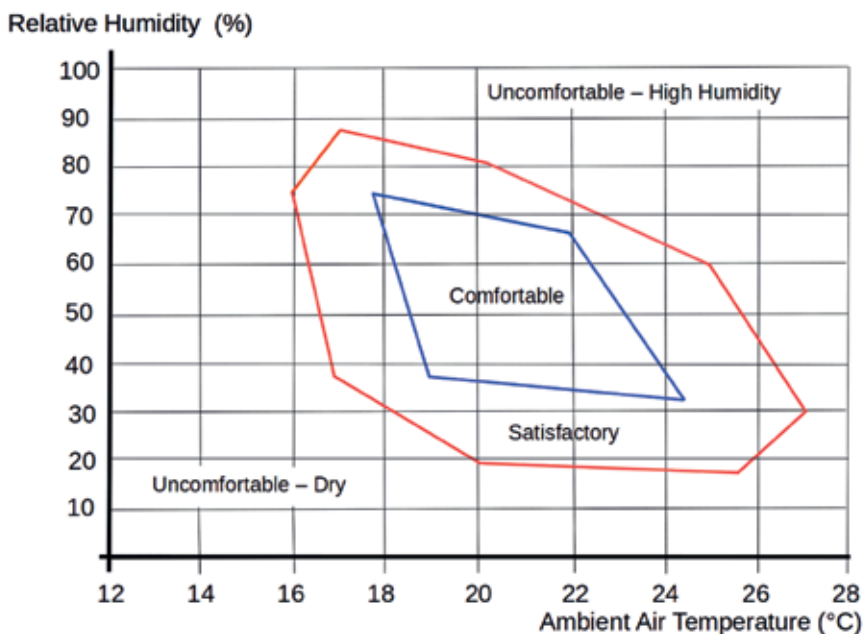


Figure 1 Graphical representation that correlates the temperature-humidity values with comfort conditions.

**2:** To run the program, on the toolbar, select the **Run** command or press the **F5** button.

**?:** Write down the **temperature**, **humidity** and **heat index** displayed on the LCD screen for the three classrooms. Describe the **comfort conditions** at the corresponding column based on the diagram above and the data you have filled in the column's **temperature** and **humidity**.



1st Hot Index Table Heat Index - Comfort Condition				
Classroom	T-Temperature	H-Humidity	HI- Heat Index	Comfort condition

**3:** Connect the LED rings to the Grove Pi port labeled with “**RINGS**” (Figure 2).



Figure 2: Arduino board & LED rings

**4:** Choose a new program. This time, always through the Geany environment, double-click the **GAIA-Workshop3.2.b.py** file and see the code in the programming environment. To run the program, on the toolbar, select the **Run** command or press the **F5** button.

**!:** The program calculates the “**Discomfort Index**” of each room at this time. The LCD screen shows the corresponding values for each classroom. Also, the corresponding number of LEDs at each ring are turned on according to the below table for each classroom on different colors.

2nd Discomfort Index Table Comfort Condition		
LED Numbers	DI – Discomfort Index	Comfort Condition
1	$DI < -1.7$	Very cold
2	$-1.7 < DI < 12.9$	Cold
3	$13.0 < DI < 14.9$	Dewiness
4	$15.0 < DI < 19.9$	Optimal
5	$20.0 < DI < 26.4$	Hot
6	$26.5 < DI < 29.9$	Very hot
7	$30.0 < DI$	Heat wave

**?:** Write down the **discomfort index** displayed on the LCD screen for the three classrooms. Based on the above table and the data you have filled in the **Discomfort Index** column, write down the **Comfort Condition** for each classroom in the corresponding column.

3rd Discomfort Index Table Discomfort Index - Comfort Condition		
CLassroom	DI- Discomfort Index	Comfort Condition

**?:** What you think is the ideal temperature and humidity for a classroom? How can you achieve the ideal values in temperature and humidity?

.....

.....



### Suggested Answers

!:

The program calculates the "heat index" in the classrooms at this time.

### Observation

The heat index (HI) is an index that combines air temperature and relative humidity, in shaded areas, to posit a human-perceived equivalent temperature, as how hot it would feel if the humidity were some other value in the shade. The result is also known as the "felt air temperature", "apparent temperature", "real feel" or "feels like".

While the heat index (HI) is being displayed at the LCD screen on the LEDs at the map is representing the classroom with the maximum heat index in red and the rest in blue.

?:

Write down the **temperature**, **humidity** and **heat index** displayed on the LCD screen for the three classrooms. Describe the **comfort conditions** at the corresponding column based on the diagram above and the data you have filled in the column's **temperature** and **humidity**.

### Answer

Once the students have recorded the values displayed on the LCD screen for each classroom, they can match the temperature and humidity values to comfort conditions based on the graph on the Figure 1.

!:

The program calculates the "discomfort index" of each room at this time. The LCD screen shows the corresponding values for each classroom. Also, the corresponding number of LEDs at each ring are turned on according to the below table for each classroom on different colors.

### Answer

The discomfort index (DI) is intended to express non-human satisfaction with the environment and prevailing conditions. It is a purely empirical indicator based on a wide range of observations and is a function of the temperature and humidity. The table corresponds to the value of the discomfort index with the LEDs lit on the rings and described it with a word.

?:

What you think is the ideal temperature and humidity for a classroom? How can you achieve the ideal values in temperature and humidity?

### Answer


The ideal Temperature-Humidity values are the ones inside the blue shape in Figure 1. If your conditions are not ideal, check which parameter is out of the blue shape. Discuss with the students about how you can interact with each of the parameters.





# Historical data

## STUDENTS: 3rd WorkSheet Level Beginner Temperature

**1:** Using the map and the components of the 1o worksheet and always through the **Geany**  **environment**, open the `GAIA-Workshop3.3.a.py` file in your school folder and view the code in the programming environment.

**2:** After disconnecting the LED connected to the “**C**” label, place the second button on the GrovePi+ port with the label “**Button 2**”.

**3:** To run the program, on the toolbar, select the **Run** command or press the **F5** button.

**!:** The program display the temperature and humidity in the classrooms up to 24 hours ago. Each time you press the “**Button**” you move one hour backwards and each time you press the “**Button 2**” you change room.

**?:** Write down a classroom historical data in the table below.

1st Historical Data Table Temperature – Humidity		
Time	Temperature	Humidity
18:..		
16:..		
14:..		
12:..		
10:..		
08:..		
06:..		
04:..		

### Main Concepts – Keywords:



Raspberry Pi, GrovePi+,  
Electroninks LED, LCD screen,  
button, LED rings, Temperature  
Sensor, Humidity Sensor,  
Discomfort index, Heat Index

### Laboratory Equipment - Preparation



- Raspberry Pi
- GrovePi+
- LCD Screen
- Button (2)
- Adaptor HDMI to VGA
- Power Supply
- Electroninks LED (3)
- Schematic
- Metal Surface
- Conductive ink
- Two terminal Cables (3)
- LED Rings



**?:** Mark the previews hours on the below graph (Figure 1) depending on the temperature and humidity you recorded in the above table.

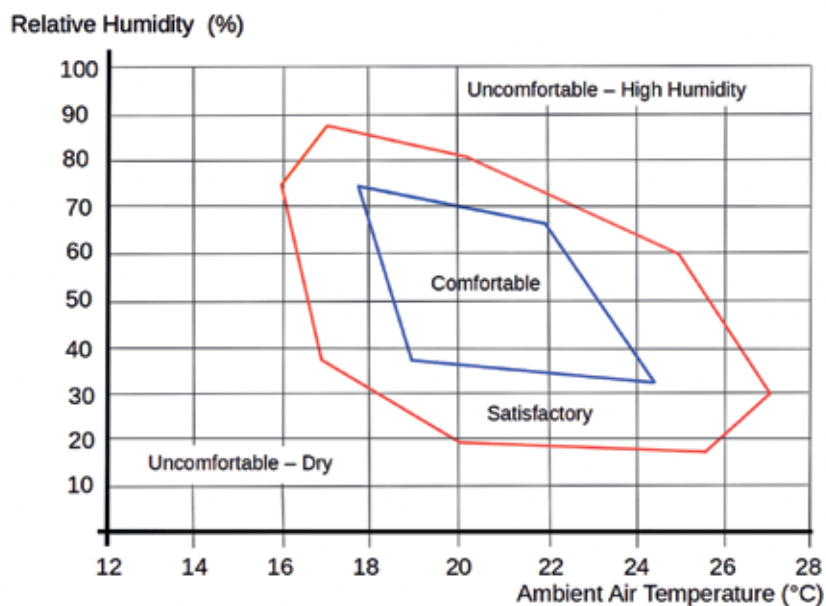



Figure 1 Graphical representation that correlates the temperature-humidity values with comfort conditions.

**4:** Open the `GAIA-Workshop3.3.b.py` file trough the **Geany**  **environment** and tun the program. The program calculates the “**Discomfort Index**” in the classrooms up to 24 hours ago. For each classroom, the LCD shows the **comfort conditions** and activates the corresponding LEDs in the LED rings.

**!:** The program calculate the “**Discomfort Index**” in the classrooms up to 24 hours ago. Each time you press the “**Button**” you move one hour backwards and each time you press the “**Button 2**” you change room.

**?:** Write down the **LEDs number** turned on for each hour and the corresponding **Comfort Condition** in a selected classroom in the below table.

2nd Historical Data Table LED Number - Comfort Condition		
Time	LED Number	Comfort Condition
18:..		
16:..		
14:..		
12:..		
10:..		
08:..		
06:..		
04:..		



**?:** Draw the squares for each hour starting from the **Very Cold ( 1 )** based on the number of LEDs you have recorded in the table above.

Heat wave - 7								
Very Hot - 6								
Hot - 5								
Optimal - 4								
Dewiness - 3								
Cold - 2								
Very cold - 1								
	04:..	06:..	08:..	10:..	12:..	14:00	16:00	18:00

**?:** Observe the time of the day (hours) during which the classroom is in **Optimal Comfort Condition**. What time of the day (hours) the classroom is in **Optimal Comfort Condition**? How long takes the classroom to reach the **Optimal Comfort Condition**?

.....  
 .....  
 .....

## TEACHERS: 3rd WorkSheet Level Beginner Temperature

### Suggested Answers

**!:** The program display the temperature and humidity in the classrooms up to 24 hours ago. Each time you press the "Button" you move one hour backwards and each time you press the "Button 2" you change room.

### Answer

*The displayed values on the LCD screen that you have recorded allow you to have an idea of the climate conditions prevailing in your classroom in the last 24 hours. The LED at the classroom which have the lowest temperature will be red and the rest of the LEDs will be blue.*

**?:** Mark the previews hours on the below graph (Figure 1) depending on the temperature and humidity you recorded in the above table.

### Answer

*By plotting table values on the graph, students can observe*

*the temperature-humidity changes that occur in the classroom during the school schedule. If you notice discontinuities, you can link them to the program and the use of each classroom.*

**?:** Observe the time of the day (hours) during which the classroom is in **Optimal Comfort Condition**. What time of the day (hours) the classroom is in **Optimal Comfort Condition**? How long takes the classroom to reach the **Optimal Comfort Condition**?


### Answer

*In this question you can see the conditions that are kept in your school during the 24 hours. The main observation you can make during the winter is the time it took for the school to come into normal conditions from the moment when the heating of the building began. Another example is that in school buildings in cold areas where heating remains open during the night, the heating conditions are kept constant throughout.*



# External & internal conditions

## STUDENTS: 4th WorkSheet Level Beginner Temperature

- 1:** From the **Geany**  **environment**, open the file **GAIA-Workshop3.4.py** from the folder named after your school and inspect the code in the development environment.
- 2:** After you disconnect the **LED** connected in port with label “**C**”, insert the second button in the port labeled “**Button 2**” of the **GrovePi+**.
- 3:** In order to execute the program, use the **Execute** icon from the toolbar or press the **F5** button on the keyboard.
- !:** This program collects data for the weather conditions in your schools area up to 15 hours ago and gives you ability to compare them with the temperature and humidity measured in your classrooms. Every time you press the “**Button**” you move one hour backwards in time, and when you press “**Button 2**” you change classroom.
- 4:** Having the measurements mentioned and comparing their values, write down the difference between external and internal relative humidity and temperature in the tables below. Circle the times of the day (hours) during which you see the maximum variation between internal and external conditions.
- !:** Keep in mind that most likely there weren't any people in the building and the HVAC of the building was turned off when the measurements were taken. Fill in the tables.

### Main Concepts – Keywords:

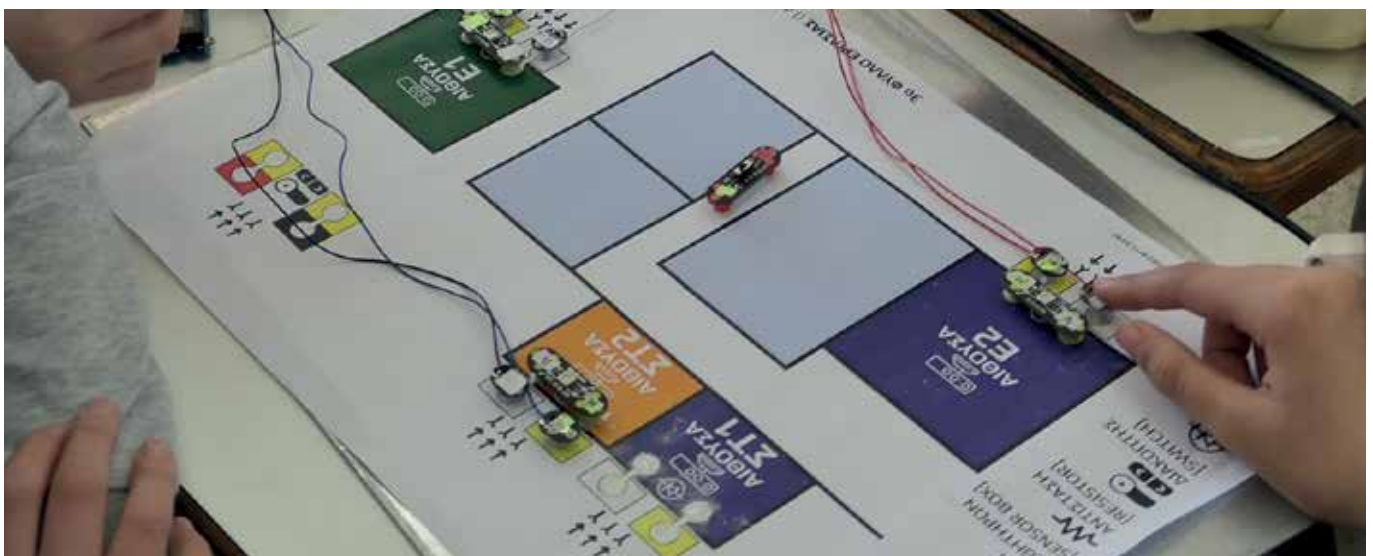


Raspberry Pi, GrovePi+,  
Electroninks LED, LCD Screen,  
Button, Temperature Sensor,  
Humidity Sensor, External  
Conditions

### Laboratory Equipment - Preparation



- Raspberry Pi
- GrovePi+
- LCD Screen
- Button
- HDMI to VGA Adapter
- Power Supply





1st External & Internal Data Table Temperature difference				
	Time	Temperature	External Temperature	Difference
Classroom .....				
Classroom .....				
Classroom .....				
Classroom .....				

**?:** Can you think of what factors might influence the results (such as orientation, geographical location, building insulation)? What factors helped the most in the improvement of the environmental conditions inside the classrooms?

.....

.....

.....



**Suggested Answers**

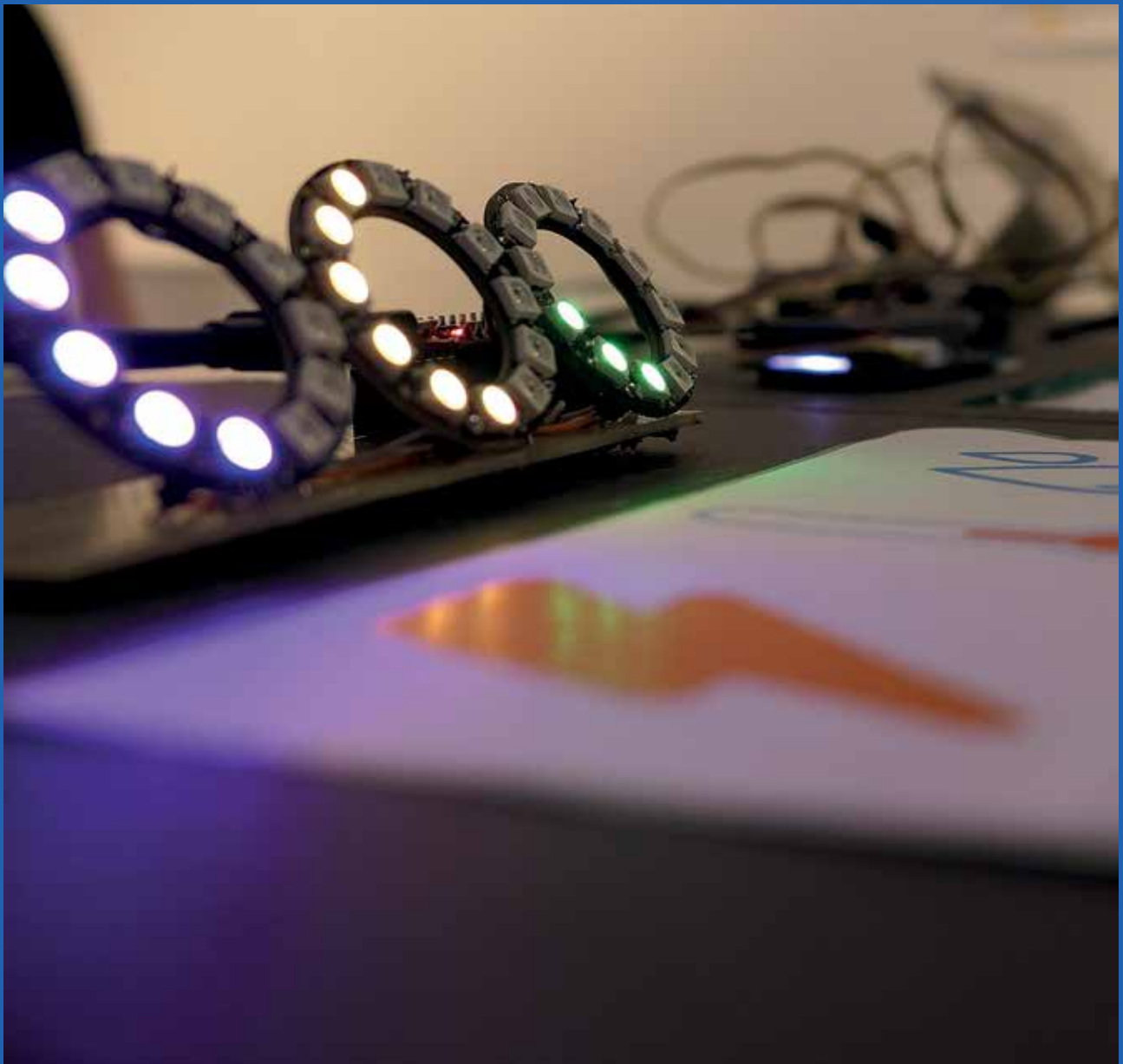
**?:** Can you think of what factors might influence the results (such as orientation, geographical location, building insulation)? What factors helped the most in the improvement of the environmental conditions inside the classrooms?

**Answer**

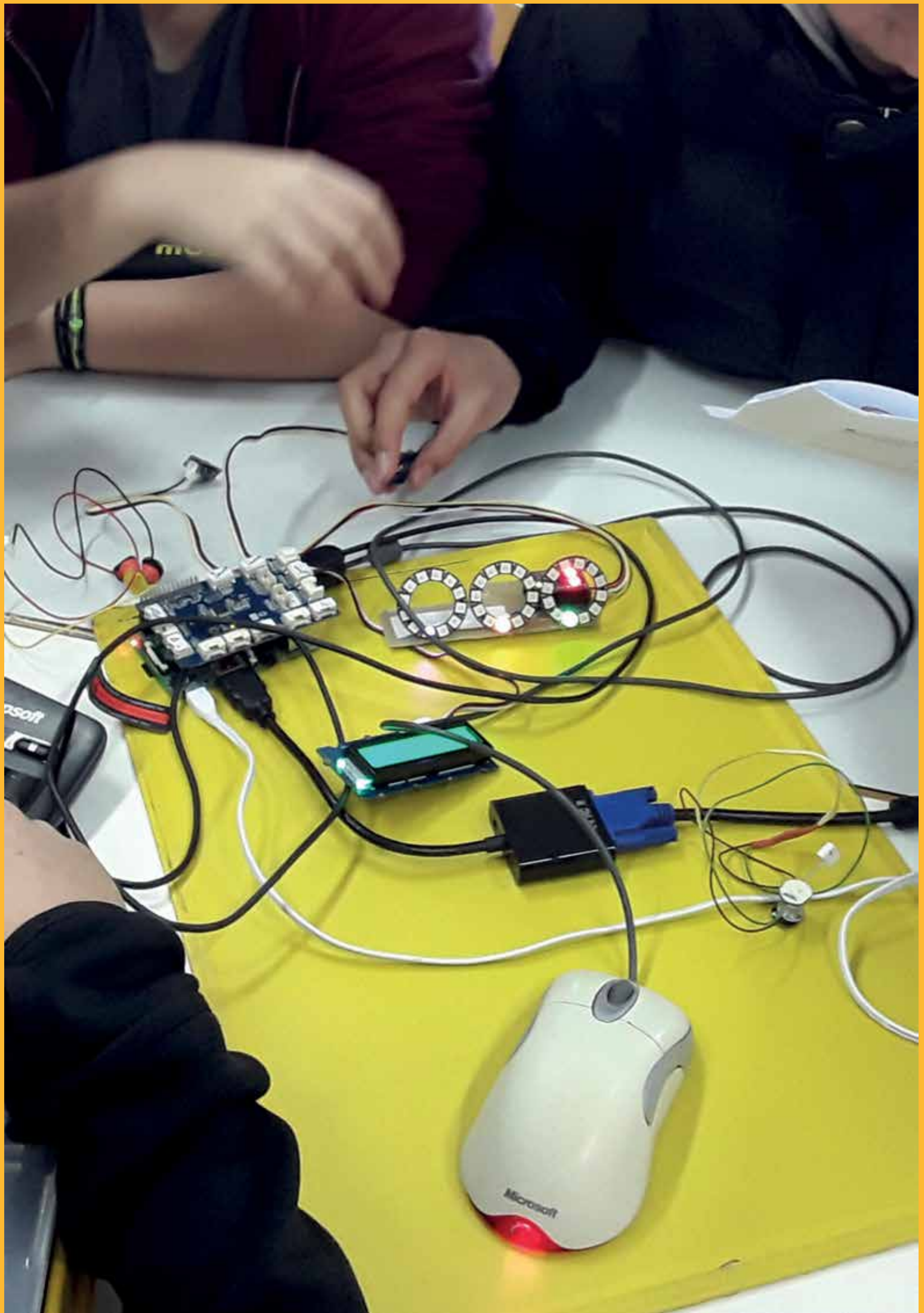
*By observing the difference between internal and external*

*temperature and humidity, you can understand how efficient your building insulation is. Insulation is influenced by the material and thickness of the wall and the windows.*

*In collaboration with your students, discuss the existing infrastructure of your school (orientation, insulation, etc) and try to justify the measurements in the tables. Also discuss other factors and their role in influencing the rate of variation of the school's internal conditions.*










# CHAPTER 7: Electrical Current

## Power usage monitoring

### STUDENTS: 1st WorkSheet Level Beginner Electrical Current

**1:** Connect the **LED Rings** cable in the port with label **Rings** on the Raspberry Pi (figure 1).

**2:** Connect the **LCD Screen** (figure 3) on the GrovePi+ (figure 2) by inserting the cable in the port with label “**Screen**”.

**3:** From the **Geany**  **Environment**, open the file **GAIA-Workshop4.1.py** from the folder named after your school and inspect the code in the development environment.

**!:** The goal of this workshop is to observe the power usage of the school building in this moment, and what can be done to reduce it.

**4:** In order to execute the program, use the **Execute** icon from the toolbar or press the **F5** button on the keyboard.

**!:** On the **LCD Screen** you can see the power usage in **Watts** for each of the phases at this moment. On the **LED Rings** the power usage is shown for each phase in relation to the maximum power usage measurement in the past four hours. The measurements are refreshed every thirty seconds, as they are measured by the sensors attached on the cables in the school's electrical panel.



Figure 1: Arduino board & 3 LED Rings.

### Main Concepts – Keywords:



Raspberry Pi, GrovePi+, Arduino, LCD Screen, Button, Three-phase grid, Appliance power, Electrical Current, Watt, Ampere, LED Rings

### Laboratory



### Equipment - Preparation

- Raspberry Pi
- GrovePi+
- LCD Screen
- Button
- HDMI to VGA Adapter
- Power Supply
- LED Rings



Figure 2: GrovePi.




Figure 3: GrovePi's LCD Screen..



**?:** How significant is the power consumption due lights in my school?  
Turn on the lights of your building the write down the total energy consumption in the following table. Afterwards, repeat the measurement after you turn off the lights in the same classrooms. What difference do you observe between the two states (lights on – lights off)?

**2:** Connect the **LCD Screen** (figure 3) on the GrovePi+ (figure 2) by inserting the cable in the port with label **“Screen”**.

**3:** From the **Geany**  **Environment**, open the file **GAIA-Workshop4.1.py** from the folder named after your school and inspect the code in the development environment.

**!:** The goal of this workshop is to observe the power usage of the school building in this moment, and what can be done to reduce it.

**4:** In order to execute the program, use the **Execute** icon from the toolbar or press the **F5** button on the keyboard.

**!:** On the **LCD Screen** you can see the power usage in **Watts** for each of the phases at this moment. On the **LED Rings** the power usage is shown for each phase in relation to the maximum power usage measurement in the past four hours. The measurements are refreshed every thirty seconds, as they are measured by the sensors attached on the cables in the school's electrical panel.

1st TABLE POWER USAGE IN WATTS				
	DRAFT	ON	OFF	DIFFERENCE
1				
2				
3				

**?:** When the lights are on, in which phase do you observe a higher power usage?

**?:** How much was the increase of power usage with lights on in relation to the total power usage of your school?

**?:** Discuss with each other and try to find out which appliances have the highest power usage in your school. Mention at least two of them in the following table:

What appliances in your school consume the most energy?

1.

2.

3.

4.

5.





**You have a quest!** Measure a few of your school's appliances to find which one is consuming the most electric energy. To measure and appliance, first deactivate all the appliances you would like to measure and write down your school's power usage. Activate the device and write down the new power usage of your school. The difference between the consumption with and without the appliance is the appliance's consumption. Repeat the process for each appliance you want to measure. Write down the results in the following table.

**2nd TABLE APPLIANCE POWER USAGE IN WATTS**

Appliance:				
	DRAFT	APPLIANCE ON	APPLIANCE OFF	DIFFERENCE
PHASE 1				
PHASE 2				
PHASE 3				

Appliance:				
	DRAFT	APPLIANCE ON	APPLIANCE OFF	DIFFERENCE
PHASE 1				
PHASE 2				
PHASE 3				

Appliance:				
	DRAFT	APPLIANCE ON	APPLIANCE OFF	DIFFERENCE
PHASE 1				
PHASE 2				
PHASE 3				



Pay extra attention to your measurements! They can be influenced by the activation or deactivation of appliances in other parts of your school.

**3rd TABLE OF ENERGY CONSUMING APPLIANCES**

Phase	Appliance	Power Usage (Watt)



According to your experiments, which are the two appliances that can influence the power usage of your school the most?



## Suggested Answers

**!:** The goal of this workshop is to observe the power consumption of the school building in this moment, and what can be done to reduce it.

## Observation

The quantities that are going to be used below are the electric **Current** (also referred to as **Current**), **Voltage**, electric **Power** and electrical **Energy**. The quantity measured by the meters located in your school is the electric Current supplied to the three phases from the grid at the time of measurement.

- Electric **Current** is the flow of an electric charge and it is measured in **Ampere(A)**.
- **Voltage** is the difference in electric potential, it is measured in **Volt(V)** and in school building in Greece it can be considered constant at **230V**.
- **Energy** is the amount of work required for a system, such as an electric appliance, to change its state. It is measured in Joule(J) but in this context it can also be measured in **kilowatt hours(kWh)**. **1kWh** is equal to **3.600.00J**.
- **Power** is the transfer rate of **Energy** per second. It is proportional to the **Voltage** at the ends of the appliance and the Current flowing through it. Power is measured in **Watt(W)**.

Out of these quantities, the most relevant for this document, are **Current** and **Energy**. **Current** is the quantity measured by the sensors installed in the electrical panels at your school. By using that measurement and assuming constant **Voltage** at **230V**, **Power** can be calculated using the formula **Power = Current x Voltage**. Power is used to calculate the Energy consumption for a given period of time by using the formula **Energy[J] = Power[W] x Time[s]**.

Pay extra attention where you will execute the experiments, because the sensors in your school might monitor only a part of it.

**?:** How significant is the power consumption due to lights in my school?

Turn on the lights of your building the write down the total energy consumption in the following table. Afterwards, repeat the measurement after you turn off the lights in the same classrooms. What difference do you observe between the two states (lights on – lights off)?

## Answer

- During recording each appliance, it is required for the state of the school building to remain unaltered. In effect, there should be no other appliances changing state except of the one being measured.
- Before initiating the recording procedure for each appliance, turn OFF the appliance and inspect the electric **Power(W)** as it is shown in the **LCD Screen** for each phase as described above. Write down each measurement in column **DRAFT** in lines **Phase 1**, **Phase 2** and **Phase 3** respectively.

- Turn ON the appliance being measured and wait for a few minutes. Inspect the measurements in **GAIA Companion App** for each phase and write them down in column **LIGHTS ON** into columns **Phase 1**, **Phase 2** and **Phase 3** respectively.
- Turn OFF the appliance being measured again. Inspect the sensor readings in the **LCD Screen** for each phase and write them down in column **LIGHTS OFF** for **Phase 1**, **Phase 2** and **Phase 3** respectively.
  - **Note:** In case the values between **DRAFT** and **LIGHTS OFF** differ, you will have to repeat the procedure to verify your measurement. The state of your school might have changed during the experiment. Differences of 25W can be ignored.
- Calculate the difference for each phase and write it down in column **DIFFERENCE**.
  - **DIFFERENCE = LIGHTS ON – LIGHTS OFF**

**?:** When the lights are on, in which phase do you observe a higher power usage?

## Answer

Observe which phase had the biggest increase.

**?:** How much was the increase of power usage with lights on in relation to the total power usage of your school?

## Answer

As a general observation, usually the appliance group with the highest power usage in school buildings is the lighting. Depending on which parts of your school are being monitored, the variation can be from normal to very significant.

**?:** Discuss with each other and try to find out which appliances have the highest power usage in your school. Mention at least two of them in the following table:

## Answer

The appliances that consume the most energy can be projectors, HVAC units, refrigerators, ovens etc.

**?:** **You have a quest!** Measure a few of your school's appliances to find which one is consuming the most electric energy. To measure an appliance, first deactivate all the appliances you would like to measure and write down your school's power usage. Activate the device and write down the new power usage of your school. The difference between the consumptions with and without the appliance is the appliance's consumption. Repeat the process for each appliance you want to measure. Write down the results in the following table.

## Answer

By following the procedure described above, you can measure which appliances consume the most energy in the tables. You can find out the phase which provides the power required by each appliance, its power usage and its name to write them down in the tables accordingly.




# Daily Historical Data

## STUDENTS: 2nd WorkSheet Level Beginner Electrical Current

**1:** Connect the **LED Rings** cable in the port with label **Rings** on the Raspberry Pi (figure 1).

**2:** Connect the **LCD Screen** (figure 3) on the GrovePi+ (figure 2) by inserting the cable in the port with label “**Screen**” and the two buttons into power “**Button**” and “**Button 2**” respectively.

**3:** From the **Geany**  **Environment**, open the file **GAIA-Workshop4.2.py** from the folder named after your school and inspect the code in the development environment.

**!:** The goal in this workshop is to observe the power usage of your school during the last 24 hours and discuss what can be done to reduce it.

**4:** In order to execute the program, use the **Execute** icon from the toolbar or press the **F5** button on the keyboard.

**!:** On the **LCD Screen**, you can see the power usage in **Watts** for each of the phases at this moment. On the **LED Rings**, the power usage is shown for each phase in relation to the maximum power usage measurement in the past four hours. Every time you press the “**Button**” the values are refreshed and you move **one hour** backwards in time. This way, you can observe the measurements in the **past 24 hours** and find out which time of day had the highest and lowest power usage.



Figure 1: Arduino board & 3 LED Rings.



Figure 2: GrovePi.



Figure 3: GrovePi's LCD Screen..

### Main Concepts – Keywords:



Raspberry Pi, GrovePi+, Arduino, LCD Screen, Button, Three-phase grid, Appliance power, Electrical Current, Watt, Ampere, LED Rings

### Laboratory Equipment - Preparation



- Raspberry Pi
- GrovePi+
- LCD Screen
- Button
- HDMI to VGA Adapter
- Power Supply
- LED Rings



**!:** The measurements shown is the power usage in Watts recorded by **one power meter in your school building for each phase**. Keep in mind that each time you press the “**Button**” you **move backwards in time** by one hour. By pressing “**Button 2**” you can change the measurements shown on the **LCD Screen** between the **total power usage on all phases** and **the power usage on each phase separately** for the currently selected hour.

**?:** Do you see a different amount of LEDs turned-on on each LED Ring? What might be the reason?

.....

**5:** Write down the measurements in the table below and circle the maximum power usage in each column.

1st TABLE POWER USAGE HISTORY DURING A DAY				
Time	Phase 1	Phase 2	Phase 3	Total Power Usage
08:..				
09:..				
10:..				
11:..				
12:..				
13:..				
14:..				
15:..				
20:..				
04:..				

**?:** During which time of the day (hour) does your school exhibit the highest power usage?

.....

**?:** Can you think of any reasons that your school exhibits the highest power usage at that specific time of the day? Is it possible to do anything to reduce it?

.....

.....

.....





### Suggested Answers

**!:** The goal in this workshop is to observe the power usage of your school during the last 24 hours and discuss what can be done to reduce it.

### Observation

*Pay attention in which part of your school you will execute the experiments as the power meters in your school might measure only a part of it.*

**?:** Do you see a different amount of LEDs turned-on on each **LED Ring**? What might be the reason?

### Answer

*Different appliances draw power from different phases. Depending on what phase each appliance is connected and its usage, the amount of turned-on LEDs changes. More turned-on LEDs means higher power usage for the respective phase.*

**?:** During which time of the day (hour) does your school exhibit the highest power usage?

### Answer

*Observe which current phase had the biggest increase.*

**?:** How much was the increase of power usage with lights on in relation to the total power usage of your school?

### Answer

*By using the information in the table and especially the Total Power Usage, write the time of the day (hour) with highest power usage and its measurement.*

**?:** Can you think of any reasons that your school exhibits the highest power usage at that specific time of the day? Is it possible to do anything to reduce it?

### Answer

*Discuss with your students each time of the day which exhibits increased power usage and identify which appliances might be active during that time period.*

*As a general observations, the highest power usage is likely to be during the first hours of the school day as the natural light is insufficient.*






# Historical Data during Recesses

## STUDENTS: 3rd WorkSheet Level Beginner Electrical Current

**1:** Connect the **LED Rings** cable in the port with label **Rings** cable in the port with label Rings on the Raspberry Pi (figure 1).

**2:** Connect the **LCD Screen** (figure 3) on the GrovePi+ (figure 2) by inserting the cable in the port with label **“Screen”** and the two buttons into power **“Button”** and **“Button 2”** respectively.

**3:** From the **Geany**  **Environment**, open the file **GAIA-Workshop4.3.py** from the folder named after your school and inspect the code in the development environment.

**!:** The goal in this workshop is to observe the power usage of your school during recesses and discuss what can be done to reduce it.

**4:** In order to execute the program, use the **Execute** icon from the toolbar or press the **F5** button on the keyboard.

**!:** On the **LCD Screen** you can see the power usage in **Watts** for each of the phases for any given 5 minute interval. Every time you press the **“Button”** the values are refreshed and you move **5 minutes** backwards in time.

This way you can observe the measurements in the **past 4 hours** and find out if the power usage of your school varies or remains constant during recesses.



Figure 1: Arduino board & 3 LED Rings.

As you can see, the way this workshop works similarly to the previous one with the exception that in this case you move backwards in 5-minute intervals instead of one hour.

**5:** Write down the measurements in the table below and answer the following questions:

### Main Concepts – Keywords:



Raspberry Pi, GrovePi+, Arduino, LCD Screen, Button, Three-phase grid, Appliance power, Electrical Current, Watt, Ampere, LED Rings

### Laboratory Equipment - Preparation



- Raspberry Pi
- GrovePi+
- LCD Screen
- Button
- HDMI to VGA Adapter
- Power Supply
- LED Rings



Figure 2: GrovePi.



Figure 3: GrovePi's LCD Screen..



1st TABLE HISTORICAL DATA DURING RECESSES				
	Phase 1	Phase 2	Phase 3	Total Power Usage
Before				
<b>Recess</b>				
After				
Before				
<b>Recess</b>				
After				
Before				
<b>Recess</b>				
After				

**?:** During recesses what is the variation of the power usage in your school building? Is it something you would expect or not? Did you expect it to increase or decrease?

**?:** If the total power usage during recesses remains constant or increases, can you think of ways to reduce it?

### Suggested Answers

**!:** The goal in this workshop is to observe the power usage of your school during recesses and discuss what can be done to reduce it.

### Observation

*Pay attention to which part of your school you will execute the experiments as the power meters in your school might measure only a part of it.*

**?:** During recesses what is the variation of the power usage in your school building? Is it something you would expect or not? Did you expect it to increase or decrease?

### Answer

*Observe the power usage just before, during and just after the recesses. In case you observe constant power usage, most likely it means that the lights in the classrooms*

*remained turned on. If you turn them off you will see a decrease in power usage. Discuss if the varying was significant or insignificant and try to think of other appliances or behaviors that influence the power usage during recesses.*

**?:** If the total power usage during recesses remains constant or increases, can you think of ways to reduce it?

### Answer

**Some suggested courses of action to implement to achieve reduction in power usage.**

- 1. Elect a student on a daily basis to be in charge of closing the lights. This way all of the students will be in the position and increase their awareness of energy consumption and cultivate that behavior.*
- 2. As an art project, create in collaboration with your students, hand-made panels positioned close to the light switches or other appliances to remind them to turn them off after using them.*







# CHAPTER 8

## Gaia Nodes

### 1 Introduction

Node-RED<sup>1</sup> is a visual tool for wiring the Internet of Things, but it can also be used for other types of applications to quickly assemble flows of services. It is a tool implemented with NodeJS (server-side JavaScript) and it is available as open source software. Node-RED's main use case is the development of Internet of Things applications, but it is a general-purpose event-processing engine. E.g., you can use it to listen to events from HTTP, WebSockets, TCP, Twitter and more and store this data in databases, without having to program much if at all. You can also use it to implement simple HTTP APIs.

Figure 1 gives an example of the Node-RED's graphical user interface and how the platform breaks systems down into simpler parts.

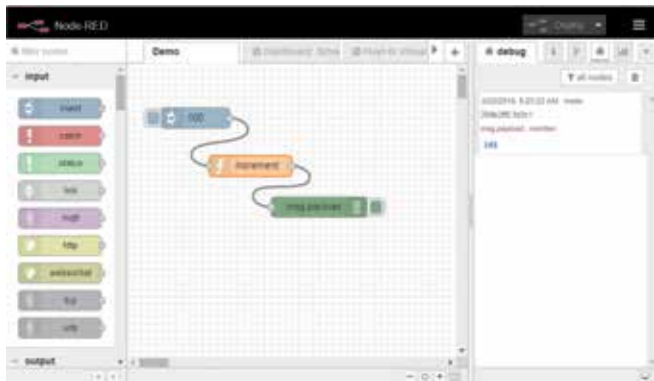


Figure 1: Example of a simple flow in Node-RED

Each of the rounded blocks on the screen represents a node, which is a visual representation of a block of JavaScript code designed to carry out a specific task. To build the demo program the user can drag into the central window a black, i.e., a node designed to output a message to other nodes. In the example, this is edited to provide the number 100 as output. Next, is dragged on and edited to define a JavaScript function that increments the number contained in the payload of the message it receives (called in the figure).

These two nodes are then wired together. Most nodes have a grey circle on their left edge, which represents their input port, and another circle on their right edge, which represents their output port. Left clicking and dragging the output port of the "node to the input port of the " node, connects the two together. The final stage consists in adding a " that prints the

message it receives in the Debug window. This is then wired to the output of the node. The process is now ready to run and once deployed will display the string 'Hello World' in the Debug box, shown on the right of the screen.

This demo program is an example of what is called a *flow* in Node-RED. The video tutorial [GaiaNode - Node-RED basics](#) in GAIA's YouTube channel introduces basic Node-RED concepts and features. Node-RED comes with a set of ready-to-use customizable nodes. You can add new nodes by searching for what you need in the nodes repository (*Menu → Manage Palette → Install*). Despite many nodes built-in in Node-RED and available in the repository, you may need a generic *function* node where to write a few lines of code for processing the incoming messages in the way you want. Node-RED can also be easily extended by developing and adding new nodes to the repository. A Gaia-Node has been developed in the framework of the GAIA project to let GAIA sensors and API be used as data sources in Node-RED.

Since Node-RED is a visual tool that allows users with minimal programming skills to rapidly assemble and deploy an IoT application, it can also be used in schools for performing simple activities with sensors and IoT data processing during class hours. Integrating Node-RED with GAIA APIs, and, optionally, local sensor kits, thus allows enriching GAIA educational activities based on the use of real data gathered from the GAIA infrastructure, while leveraging a tool supported by the open source community and a rich documentation (see additional resources).

### 2 Gaia-Node

The Gaia-Node<sup>2</sup> plug-in/module for Node-RED is a set of *nodes* that allows interacting with the GAIA platform to access available resources. The nodes essentially let you query the GAIA IoT platform and retrieve the measurements gathered by the sensors in GAIA's schools (given that you have the credentials to do so).

#### 2.1 Installation and documentation

First, a user needs a valid GAIA account (teacher or student) and a short training to understand the basics. To install the GaiaNode module just type `node-red-contrib-gaianode` and click install. This action will add a set of nodes able to interact with the GAIA platform (e.g., query values, push values,

<sup>1</sup> <https://nodered.org/>

<sup>2</sup>



etc.). Please first go through the README file, to avoid typical installation errors. The plugin is under development, so the README file may be get updated.

### 3 Educational Activities

A wealth of educational activities can be designed leveraging Node-RED. In GAIA's YouTube channel, you can find a set of activities designed for schools based on Node-RED. A set of video tutorials has also been produced to support teachers in instructing students.

**Table 1 Video tutorials**

<a href="#">Node-RED basics</a>	Learn the basics of the Node-RED tool, understand the message flow, be able to deploy a simple flow and use the debug panel
<a href="#">Read a sensor</a>	Create a flow to retrieve the latest value read by a sensor, given the resource id, and display it in the debug panel
<a href="#">Virtual Sensor</a>	Create a virtual sensor using the GAIA BMS and send new measurements using Node-RED.
<a href="#">LCD Screen</a>	Read a value from a sensor and/or from the platform and display it on a connected LCD screen.
<a href="#">Web Dashboard - 1 Basics</a>	Create a web dashboard e.g., displayed on a tablet/screen in the hall of the school
<a href="#">Web Dashboard - 2 Rotating Dashboard</a>	Create a web dashboard e.g., displayed on a tablet/screen in the hall of the school

### Resources for Gaia-Node

- Gaia-Node documentation <https://github.com/buddino/node-red-contrib-gaianode/blob/master/README.md>
- Gaia-Node activities <https://drive.google.com/file/d/1eWJqTGyudj6RMpIfAeHt9eA5pR6l8H8G>
- <https://raw.githubusercontent.com/buddino/node-red-contrib-gaianode/master/README.md>
- <https://nodered.org/>
- <http://noderedguide.com/>

### Installation

You can install gaianodes directly using the Node-RED editor, i.e., the web interface. To do this, select *Manage Palette* from the menu (top right), and then select the *install* tab in the palette. You can now search for new nodes to install; just type *gaianode* and select *node-red-contrib-gaianode*. After a while, the new nodes will appear in the left panel under the group *Gaia*.

Otherwise, you can install gaianodes within your user data directory (by default, %HOME%/.node-red) by typing:

```
cd %HOME%/.node-red
npm install node-red-contrib-gaianode
```

### Available Node-RED nodes

- **LatestValue**: retrieves the latest value (with some additional information) of the resource identified by the id given to the node
- **Summary**: retrieves the summary (latest value, averages, min, max, latest values at different granularities) of the resource identified by the id given to the node
- **RealTime**: connects to the WebSocket output of the GAIA Platform allowing to receive real time unprocessed data as injected into the platform by the sensors
- **Notifications**: connects to the WebSocket output of the GAIA Platform allowing to receive real time unprocessed data as injected into the platform by the sensors
- **ListResources**: list the resources (sensors) associated with the given site id
- **Timerange**: retrieve the values in the specified time window at the specified granularity (5min, hour, day, month) for the given resource
- **PushValue**: push a value into the GAIA Platform (a valid virtual sensor id is needed, see BMS manual for information)
- **Uri2Id**: convert the given URI to a numeric resource id to be used with other GaiaNodes

### LatestValue



### Parameters

- **Resource id**: numeric id or a mustache-style string (e.g. {{msg.payload}})
- **Gaia Server**: a configured GAIA Server

### Output

- **payload**
  - **uri**: Literal identifier of the resource
  - **uom**: The Unit of measurement
  - **latestTime**: The Timestamp of the measurement in milliseconds (UNIX time)
  - **latest**: The latest value measured
  - **latestMin5**: The averaged value during the latest 5 minutes
  - **latestMin60**: The averaged value during the latest hour
  - **latestDay**: The averaged value during the latest day
  - **latestMonth**: The averaged value during the latest month
- **topic**: The id of the queried resource

### Details

The queried resource can be configured writing directly its id (e.g. 155076) or by using mustache-style tags (e.g. {{msg.payload.id}} or {{msg.payload.resource}} for accessing the fields of the input message)



## Summary

Retrieves the summary for the given resource id from the GAIA Platform

## Output

- **payload**
  - **uri**: Literal identifier of the resource
  - **uom**: The Unit of measurement
  - **latestTime**: The Timestamp of the measurement in milliseconds (UNIX time)
  - **latest**: The latest value measured
  - **minutes5**: Array containing the latest 48 values at 5 minutes interval
  - **minutes60**: Array containing the latest 48 values at one hour interval
  - **day**: Array containing the latest 48 values at one day interval
  - **month**: Array containing the latest 48 values at one month interval
  - **min**: Minimum value of the latest (5 minutes, hour, day, month)
  - **max**: Maximum value of the latest (5 minutes, hour, day, month)
  - **max**: Mean value of the latest (5 minutes, hour, day, month)
- **topic**: The id of the queried resource

## Timerange

Retrieves the values of the given resource within the specified time window.

### Parameters

- **Resource id**: numeric id or a mustache-style string (e.g. `{{msg.payload}}`)
- **From**: a date and a time (if not provided by the input message)
- **To**: a date and a time (if not provided by the input message)
- **Granularity**: the interval between queried values (if not provided by the input message). Possible values are: *5 minutes, 1 hour, 1 day, 1 month*
- **Gaia Server**: a configured GAIA Server

## Input

- **from**: date and time in the format *2010/08/17 12:09:36* or as UNIX timestamp
- **to**: date and time in the format *2010/08/17 12:09:36* or as UNIX timestamp
- **granularity**: Possible values are: *5 minutes, 1 hour, 1 day, 1 month*

## Output

- **payload** is a dictionary whose key are the ids of the requested resources (*currently only one resource at a time is supported*)

- **average**: Average of the values in the time range
- **summary**: Sum of the values in the time range
- **data**: Array of measurements. Each element contains:
  - **reading**: measurement value
  - **timestamp**: UNIX timestamp of the reading (milliseconds)
  - **topic**: The id of the queried resource

## PushValue

Send a value to the specified resource (Virtual sensor) on the GAIA platform. Virtual sensors can be created using the GAIA [BMS application](#).

### Parameters

- **Resource id**: numeric id or a mustache-style string (e.g. `{{msg.payload}}`)
- **Gaia Server**: a configured Gaia Server

## Creation of a virtual sensor

You can refer to the [BMS - User Guide](#) pages 14-15. Currently the best way for retrieving the resource id of a Virtual Sensor is to open its page (by clicking on the sensor) and take the id from the URL shown in the browser (e.g. [http://bms.gaia-project.eu/#/page/sensor/view/\\*\\*1000565\\*](http://bms.gaia-project.eu/#/page/sensor/view/**1000565*) )

## Uri2Id

This node converts the textual URI of a resource to a numeric resource id to be used into the other GAIA nodes.

### Parameters

- **Resource id**: textual URI or a mustache-style string (e.g. `{{msg.payload.uri}}`)
- **Gaia Server**: a configured GAIA Server

## Output

- **payload** contains the numeric id of the resource identified by the provided URI

## ListResource

Retrieves the list of available resources given a site identifier

### Parameters

- **Site id**: numeric id or the site you want whose resources will be listed or a mustache-style string (e.g. `{{msg.site}}`)
- **Gaia Server**: a configured GAIA Server

## Output

- **payload** contains the numeric id of the resource identified by the provided URI



- o **resources**: An array of resources (sensors in this case) each one composed by:
- **resourceId**: The numeric id of the resource
- **uri**: The URI of the resource
- **name**: The name of the resource
- **isa**: the type of resource (e.g., sensor)
- **property**: Measured property (e.g., Temperature, Luminosity)
- **uom**: Unit of measurement (e.g., Wh, A, Raw)
- **options** An array of key-value couple. The key is the URI of the resource and the value id the numeric id. This is useful for creating selection widgets (e.g., for letting the user choose in the GUI the resource to be read)

## RealTime

**Experimental** Connects to the GAIA platform to receive real time measurements as soon as they are pushed to the platform.

## Parameters

- **Path**: the path of the school/area to monitor (e.g. *ROOT.GAIA.GROUPS.155076* where 155076 is the site id of the school)
- **Gaia Server**: a configured Gaia Server

## Output

The message received by the platform are forwarded in real-time without post processing (e.g. cleaning, aggregation):

- **payload**:
  - o **timestamp**: The Timestamp of the measurement in milliseconds (UNIX time)
  - o **resourceUri**: URI of the resource
  - o **value**: Value read by the sensor

## Notifications

**Experimental** Connects to the GAIA Recommendation engine and listen for real time notifications

## Parameters

- **Site id**: numerid id or the site you want whose resources will be listed or a mustache-style string (e.g. `{{msg.site}}`)
- **Gaia Server**: a configured Gaia Server

## Output

- **payload**: the textual recommendation sent by the engine
- **notification** contains:
  - o **timestamp**: Timestamp of the event (UNIX time)
  - o **school**: Identifier of the school the rule belongs to
  - o **area**: Identifier of the area to which the rule is linked
  - o **ruleClass**: Name of the class of the rule

- o **ruleName**: Name of the rule (the instance of the rule)
- o **ruleId**: Id of the rule
- o **values**: Object containing a snapshot of the values when the rule has been fired
- o **description**: Brief description of the rule
- o **suggestion**: Same content of payload
- o **type**: The type of message received (e.g., info, alert)

## Configuration

To use the Gaia Nodes, you must create a Gaia Server (one server can be used by multiple nodes). From the configuration of first Gaia node you insert, you must choose one valid Gaia server. If you do not have configured any server yet, click on the edit button.

The Gaia server configuration window will open and you just have to enter your credentials and press *Add*. From this moment and on, you will find this server in the dropdown menu of each Gaia node.

## Details

Parameters can be configured in the node configuration typing its id (e.g. 155076) or, where specified, by using mustache-style tags (e.g. `{{msg.payload.id}}` or `{{msg.payload.resource}}`) for accessing the fields of the input message). It is also possible to configure the timestamp associated with the value by using the *Time* field in the node configuration; you can use mustache-style tags for accessing the field of the message to be used as timestamp (e.g., `{{msg.payload.timestamp}}`).

## More online resources:

Mustache <https://mustache.github.io/>  
 SparkWorks API <https://api.sparkworks.net/swagger-ui.html>  
 GAIA Android App <https://play.google.com/store/apps/details?id=eu.gaiaproject.android.companion>



# Annex

