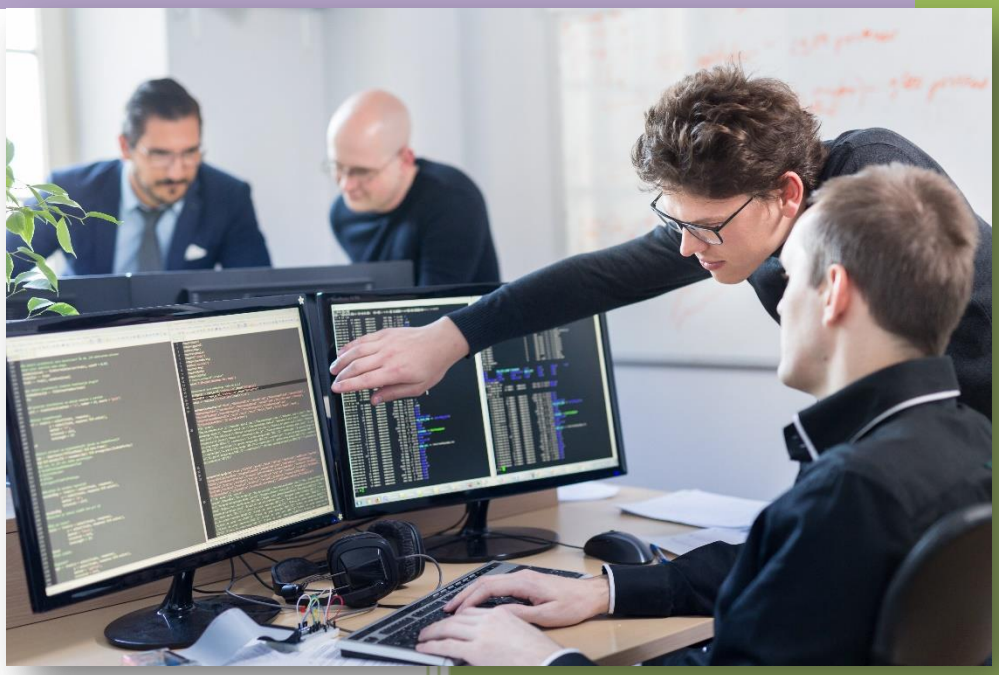
Co-funded by the
Erasmus+ Programme
of the European Union

MASTER CLASS

INTRODUCTORY CLASS IN PYTHON TARGETING GAME DEVELOPMENT



Project "Developing Innovative Science Outreach for Vocational Education to Encourage STEM Careers and Education", ref. no. 2017-1-BG01-KA202-036327

Published 2019

DISCOVER PROJECT

Co-funded by the Erasmus+ Programme of the European Union

Introduction

HOW TO USE THIS RESOURCE

This Master Class is an on-campus learning and orientation experience for high school students, combining visits to computer labs with informative interactive talks on scientific topics related to computer programming and digital games. The theme ensures that the training will be relevant to young people's lives and interests. The training activities should be planned so that they take place within the university and use available labs or other equipment.

Co-funded by the Erasmus+ Programme of the European Union

Activity concept and lesson plan

THEME OF THE CLASS	Introduction to the Python programming language and basic game programming concepts.
LEVEL OF DIFFICULTY / AGE OF STUDENTS	This Master Class targets students of upper secondary education that already have some experience in programming so that they can easily understand the basic programming concepts of Python used in the memory game.
REQUIRED PRIOR KNOWLEDGE	Students should have previous experience in using variables, if-then-else statements, loops and functions in any programming language so that they can easily understand how Python supports those programming concepts.
TIME REQUIRED FOR IMPLEMENTATION	2 to 3 academic hours
INSTRUCTORS	Ideally, university staff with experience in teaching introductory programming courses can serve as instructors (preferably using the Python programming language). The task can be easily performed by graduate students or students in more advanced stage of their studies, provided that they have received prior instruction.
KNOWLEDGE GAINED AND COMPETENCIES DEVELOPED - STUDENTS	Participating students will gain knowledge and understanding of basic programming concepts in Python: • boolean, integer and string variables • functions and local vs global variables • lists • boolean and arithmetic expressions • functions • objects and methods General knowledge and understanding of event-based programming concepts (event-based programming is the underlying programming model used in all modern graphical interfaces and also related to the management of input from remote devices as well as I/O devices connected to a computer): • Events and callbacks • Event loop Students will develop basic skills in programming.

Co-funded by the Erasmus+ Programme of the European Union

KNOWLEDGE GAINED AND COMPETENCIES DEVELOPED – SCHOOL TEACHERS

Assuming that the accompanying teachers are computer science teachers that already teach programming, they may learn the syntactic rules of Python supporting procedural programming as well as object-oriented and event-based programming paradigms (if they do not yet know them).

Accompanying teachers will also develop skills to use the CodeSkulptor platform http://www.codeskulptor.org so that they could possibly use it in their classes.

Finally, the teachers will develop their didactical competencies to teach programming.

KNOWLEDGE GAINED AND COMPETENCIES DEVELOPED - UNIVERSITY STAFF OR UNIVERSITY STUDENTS

University staff will improve their skills to both teach and communicate science.

To effectively support this Master Class, any involved university students will also develop their programming skills in Python and understand how Python could be used as an introductory programming language through the development of self-contained examples (projects) in the form of mini-games.

MATERIALS NEEDED FOR IMPLEMENTATION OF THE ACTIVITY

Multimedia projector (or TV set) and a computer with Internet connection to show to the participants the code of the memory game used in this Master Class.

Whiteboard may also be useful to explain the presented code with schematic diagrams.

Computers with Internet connection where participants can work in groups to use and remix the presented code. The students will use the CodeSkulptor platform http://www.codeskulptor.org, which provides a simple GUI library that can be used to develop simple graphics and digital games using graphical elements. CodeSkulptor is available online so there is no need for installation.

BREAKDOWN OF ACTIVITIES

The approach is based on a ready-made game that is initially presented to the students along with information about the basic programming constructs of the Python language. The initial presentation includes the key concepts related to game programming, such as events, callback functions and event loop. Following the initial presentation, the visiting students and the accompanying teachers are invited to work on possible extensions of the initially presented memory game and finally present their remixes in a final session that promotes self-reflection.

This Master Class is thus divided into 3 parts:

1. Informative talk/Lecture: This part should take place in a lecture room or amphitheatre of the organizing university so that the students can get a taste of what it is being a university student. This part starts with a welcome and a short ice-breaker activity aiming at discussing students' previous experience with programming and playing digital

Co-funded by the Erasmus+ Programme of the European Union

games. Then, the memory game is presented, students are invited to play it, and finally its code is presented.

2. Hands-on activity /Lab visit: This part should take place in a computer room, so the first step is to move from the lecture room to the computer lab in order to follow the activity. In particular, the steps of the second part are the following:

Step 1 – If they are not already in a computer lab, the students and their accompanying teachers move to a computer lab where the lab session will take place.

Step 2 - During this phase, the students are divided into groups of 2-4 persons (depending on the number of available computers and the number of participating students).

Step 3 – The groups are invited to make certain changes in the code of the memory game. They could either develop their own remixes or perform specific tasks set by the instructor.

Step 4 – The university staff and students involved in the Master Class, play the role of facilitators to help the student groups develop the selected changes/remixes. This role could also be played by the accompanying teachers.

3. Self-reflection on the part of students: This part should take place in a lecture room or amphitheatre, possibly the same one that was used for the first part of the Master Class. Consequently, the first step is to move from the computer lab back to the lecture room or amphitheatre. There, each group in turn presents its version of the memory game that was developed in part 2. They are also invited to report on the difficulties they have faced and how they solved problems that have arisen. After the students' presentations, the instructor closes the meeting providing additional links and information for those that want to learn more on the topics of the Master Class (Python programming and digital games).

USEFUL LINKS TO RESOURCES

The code of the memory game used in this Master Class is available online from:

http://www.codeskulptor.org/#user44_VlbTLt8MzXB6GmB.py

From this link it is also possible to run the code in any web browser.

Based on the above initial code, many possible extensions can be developed.

It is possible to adjust the whole Master Class to other examples of games or simulations developed in CodeSkulptor. Such an interesting alternative, which also provides excellent possibilities to talk about complexity and patterns is the Game of Life that is available at:

http://www.codeskulptor.org/#demos-game_of_life.py

Co-funded by the Erasmus+ Programme of the European Union

SUGGESTED FURTHER READING

For more information about CodeSkupltor and introductory material about Python: http://www.codeskulptor.org/docs.html#tabs-Python

SOURCES USED TO DEVELOP THE RESOURCE

This resource is based on information about CodeSkulptor and indicative student projects that are available at: https://www.theseus.fi/bitstream/handle/10024/81687/Ajayi_Richard.pdf?sequence=1

Co-funded by the Erasmus+ Programme of the European Union

Background knowledge sheet

EDITORS

Nektarios Moumoutzis (Technical University of Crete, Greece)

INTRODUCTION TO THE TOPIC

Most software tools available today provide powerful scripting languages to enable flexible customization and rich interactive content development by end-users. In this respect, knowledge of computer programming concepts is nowadays necessary for most knowledge workers, including scientists and engineers.

Consequently, many higher education departments have included introductory programming courses in their curricula. Many countries extend their curricula in secondary or even primary education to address the development of basic programming skills. The importance of computer programming has received even more attention through computer coding campaigns such as the The Hour of Code and Europe Code Week. Informal learning opportunities are also offered in many countries following the organizational approach of Coding Clubs.

However, in many cases, the introductory programming courses follow the traditional approach of presenting the features a specific programming language one by one along with artificial examples and small programming exercises. An alternative, more engaging way to promote introductory computer programming knowledge is through the use of meaningful examples and coding projects such as digital games. Digital games are very popular among young people and can offer engaging learning experiences if framed within an appropriate pedagogical framework (game-based learning or learning through game development). This is the approach taken in this Master Class.

When following this Master Class, the participating students will have the opportunity to understand the basic features of the Python programming language while at the same time they can understand the main building blocks and mechanics of digital games.

Python has recently been introduced in vocational training curricula (professional lyceums) as well as in upper secondary education in Greece and other Countries. Consequently, there is a growing popularity for the Python programming language in education. This trend resonates with the use of Python as an embedded scripting language in many platforms and applications, as well as its growing popularity in scientific applications including, data science (https://en.wikipedia.org/wiki/List_of_Python_software).

Therefore, it is important to facilitate the introduction of secondary students to Python programming in an engaging way that will help them understand the basic features and merits of this language. The approach adopted in this Master Class is to use simple digital games that can be explored and remixed using a web-based Python programming environment (CodeSkulptor).

Co-funded by the Erasmus+ Programme of the European Union

IMPORTANCE TO DAILY LIFE/ ECONOMY /SOCIETY

This is an introductory programming course. Such courses (including learning activities of short duration) have received much attention the recent years. This is mainly due to the ubiquitous use of computers, the proliferation of the so called cultures of participation (related to digital platforms that promote digital creativity), end-user programming (platforms that enable users create and code their own digital artefacts) and end-user software engineering (systems that promote co-creation of digital artefacts between end-users and software developers). In all these cases, computer programming is a means to create interactive digital artefacts and promote creative expression and deeper understanding of digital technologies.

DETAILED PRESENTATION ON THE TOPIC

DEFINITION OF TERMS

Object-oriented programming

Unlike procedure-oriented programming, which is organized around *functions*, object-oriented programming refers to a programming language organised around *objects* that combine data and functionality. Objects may contain data, in the form of fields (often referred to as *attributes)* and code, in the form of procedures, (often referred to as *methods).* An object's methods can access and often modify the data fields of the object with which they are associated. The program includes a variety of objects that interact with each other. Many of the most widely used programming languages (such as C++, Object Pascal, Java, Python, etc.) support object-oriented programming to a greater or lesser degree.

Event-based programming

Event-based programming is the underlying programming model used in all modern graphical interfaces and also related to the management of input from remote devices as well as I/O devices connected to a computer

Scripting language

A scripting language is a programming language that is interpreted by another program at runtime rather than compiled by the computer's processor.

THE MEMORY GAME

The code of the memory game used in this Master Class for introducing secondary students in Python is available at: http://www.codeskulptor.org/#user44_VlbTLt8MzXB6GmB.py.
This code demonstrates most of the basic features of Python:

- Use of libraries (import statements in lines 5-6)
- Use of comments in program code (lines 1-3, 10, 23, 52, 66, 69, 74, 78).
- Definition of functions as follows:
  - new_game() - this function is used to initialize global

Co-funded by the Erasmus+ Programme of the European Union

variables when starting a new game. These global variables include the list of cards used in the memory game, the presentation characteristics of the cards (width, height, their visibility as the use is clicking on them represented in the list 'exposed'). A new game is started when the applications starts (function new_game() is called in line 67) or whenever the player clicks on the button with the label 'Restart' (new_game() is registered as the callback for clicking this button in line 71).

- mouseclick(pos) – this function is defined in lines 24-50. It is used to handle mouse clicks. This is a callback function that is registered in line 75. When called, this function has access to the x,y coordinates of the mouse click as a list of values in input parameter 'pos'. The code of this function is organized into three sections (if-elif statement) depending on the number of previous clicks counted by variable 'counter'. This way it is possible to handle how the memory cards are exposed (revealed) to the player. The player is able to see at most two cards in the same time. If the two cards have the same symbol (letter) they remain exposed for the rest of the game and the player should find another pair of identical cards. If not, the cards are hidden again and the player has to find a pair of identical cards again while trying to remember the previously exposed cards.
- draw(canvas) – this function is defined in lines 53-64. It is called in each iteration of the main game loop. It is callback function that is registered in the game loop at line 76.

- Use of main program commands (lines 67-79). In the main programe the new_game() function is called to initialize global variables and then a CodeSkulptor simplegui fame is created (line 70) where a 'Restart' button is added (line 71) that is used used to initialize the game when clicked by registering the new_game() function as a callback (line 71). Furthermore, a label is added in the frame (line 72) that is used to count the number of times the player has played ('Turns = 0'). In line 75 the mouseclick(pos) handler is registered and in line 76 the main game loop handler (i.e. the draw(canvas) function). Finally, the last command of the main program is frame.start() to start the main game loop.
- Use of local and global variables. Variables could be simple variables storing strings, numbers or boolean values or lists (such as 'cardList', 'exposed'). Initializing lists in various ways is demonstrated in lines 13, 16, 19 and random shuffling of list elements is demonstrated in line 15. Accessing of list elements uses the [] notation and also includes accessing elements with negative indices to count from the end of a list (lines 42-44).
- Use of objects and object methods/properties using the dot(.)

Co-funded by the Erasmus+ Programme of the European Union

operator (such as in lines 70-79).

Use of loops (lines 57-59, 61-64).

SCIENCE COMMUNICATION GUIDANCE TO INSTRUCTORS

- Don't be "just another teacher" – don't just teach. Share with the students your passion for programing. This would typically entail telling them how you got into programming and why you like doing it.

- Establish a more personal connection with the students. This would typically entail telling them something about yourself or making jokes during the activity.

- Start by asking the students about their prior programming knowledge in order to link the new knowledge with what the students already know. For example, if most of the students have experience in programming with Scratch (http://scratch.mit.edu), the instructor should describe the presented code by making analogies with Scratch (e.g. the way lists are declared and used in Scratch and in Python).

- Present all the details regarding the memory game code that are described in the 'Detailed presentation' field above from a perspective that emphasizes the way the code is executed.

- Try to make students curious about other programming languages or about programming other games.

Co-funded by the Erasmus+ Programme of the European Union

Lecture planning sheet

GOAL	The goal of the informative talk is to make the students feel comfortable in the university environment, report and reflect on their prior knowledge on computer programming, and improve their knowledge and skills in programming. This is achieved at this stage by helping them understand the code of the memory game. The students will be invited to revise this code in the second part of the Master Class (during the hands-on activity).
SETTING	In principle, the informative talk can accommodate a large number of students. Any limit to the number of students will depend on the second part of the Master Class, which has to be implemented in a computer lab. During that second part the students will be organized in groups. The duration of this part of the Master Class can range from 30 to 60 minutes depending on how long the explanation of the code of the memory game will take considering the prior knowledge of the students.
LOCATION FOR THE TALK/LECTURE	Lecture room
POSSIBLE INVOLVEMENT OF UNIVERSITY STUDENTS IN THE ACTIVITY	University students may undertake the responsibility to present the code while university staff could coordinate the activity and the discussion, especially in the beginning of the activity when the secondary students are invited to report on their previous knowledge on computer programming.

Co-funded by the Erasmus+ Programme of the European Union

TIMING & RUN-DOWN

Phase no.	Description of phase	Time allocated
1	**Welcome note; clarification of the aim of the Master Class**	3-5 min
2	**Ice breaker** For example, the instructor could invite the students to work in pairs to find 2-3 things that make digital games engaging and differentiate them from traditional games or ask them to briefly describe their favourite digital game.	3-7 min
3	**Presentation of the memory game. Volunteers are asked to play it.**	2-5 min
4	**The participating students are invited to answer the following questions: How does this game work? What happens when you click the 'Restart' button or a memory card? How could you develop this game with the programing language/platforms that you already know?**	5-12 min
5	**Presentation of the memory game code** *See details presented in the 'Background Knowledge Sheet'. Volunteer university students may undertake the role of presenters here.*	15-25 min
6	**Closing words** *The instructor briefly presents what will follow and what kind of remixes the students will be facilitated to do in their activity in the computer lab.* *Students move to the computer lab.*	5 min

Co-funded by the Erasmus+ Programme of the European Union

Hands-on activity/experiment planning sheet

GOAL

The goal of the hands-on activity is to allow the students to develop their own version of the memory game by implementing possible revisions and extensions to the code presented during the first part of the Master Class. The students should be facilitated in this process to the extent necessary.

SETTING

The activity takes place in a computer lab where the students work in groups of 2-4 persons per group, depending on the available space.

Volunteer university students or the teachers accompanying the students may play the role of facilitators. Each facilitator should support at most 2 groups of students.

LOCATION AND EQUIPMENT

The hands-on part of the Master class is expected to take place in a computer lab that is usually used by university students so that the secondary students can experience the life of university students. The computers in the lab should have an active Internet connection so that CodeSkulptor can be used for accessing and modifying the code of the memory game.

POSSIBLE INVOLVEMENT OF UNIVERSITY STUDENTS IN THE ACTIVITY

As already described above, this activity is designed to involve university students as facilitators. In case of lack of volunteers, this can be done by university staff. However, the involvement of university students is highly desirable as it will help the secondary students feel more comfortable and be able also to discuss with them other things beyond programing (e.g. things related to students' life).

CONTENT OF THE HANDS-ON ACTIVITY

During this activity, the secondary students will be invited to modify the code of the initially presented memory game. They will also be given several ideas for extensions that they could implement or they could develop their own ideas.

Indicative extensions that should be suggested to the participating students are the following:

- Change the representation of the memory cards (e.g. different letters, numbers, other symbols).

Level of difficulty for this extension: LOW

- Change the visual appearance of cards (e.g. colours, fonts…).

Level of difficulty for this extension: LOW

- Change the number of cards in a group that should be revealed in order to proceed. Initially, each group consists of two identical cards. This can be changed so that each group could have 3 or 4 cards, thus increasing the difficulty of the game.

Co-funded by the Erasmus+ Programme of the European Union

Level of difficulty for this extension: INTERMEDIATE

- Change the memory cards grid from a sequence to a rectangle (e.g. 4X4 for 16 cards).

Level of difficulty for this extension: HIGH

- Depending on the extension selection, the code could be further modified to give different options to the user that can be selected with certain buttons. E.g. select the desired representation of memory cards, the visual appearance of cards, the number of cards in a group or the grid size. All the above options can be controlled with new buttons in the control pane of the game in CodeSkulptor.

Level of difficulty for this extension: INTERMEDIATE

TIMING & RUN-DOWN OF THE HANDS-ON ACTIVITY

Phase no.	Description of phase	Time allocated
1	**Organization of groups so that in each group there is a balance between students that have previous experience in programming and students that do not have strong programming skills.**	2-5 min.
2	**Brief intro to the CodeSkulptor interface**	2-5 min.
3	**Presentation of possible extensions**	5-10 min.
4	**Students work in groups to develop their extensions** *University volunteer students act as facilitators.*	20-40 min.
5	**Feedback and self-reflection** *The groups present in turn their creations and report on the problems they have confronted and the solutions they have found. This final reflection phase can be done back in the lecture room or amphitheatre that was used for the initial informative talk.*	30-40 min.
6	**Conclusions and farewell** *The instructor concludes*	5 min.

Co-funded by the Erasmus+ Programme of the European Union

Annex I: Knowledge Resource

The code of the memory game used in this Master Class is available at: http://www.codeskulptor.org/#user44_VlbTLt8MzXB6GmB.py.

Co-funded by the Erasmus+ Programme of the European Union

Annex II: Co-Creation

University Students

Selection	
The following university students can be involved in the design and delivery of the activity: - Computer Science students, in any year of studies, as long as they are knowledgeable in programming in general and Python in particular. - Students should be selected by the faculty member responsible for the activity and should have worked with this faculty member before (in class or in educational outreach activities). The selected students should stand out for their science communication skills rather than their excellence and academic achievement per se.	
Role (in order of relevance)	**Guidance**
Pedagogical co-designers of learning, teaching and assessment; facilitators in hands-on and lab experiments	The selected university students: - should work together with high school students during the practical activity and help them develop extensions of the proposed program - should participate in the assessment of student performance during the Master Class and in the evaluation of the effectiveness of the training - should also be actively engaged in the self-reflection phase, staying with the student team in which they worked.
Mentors of SE VET students	The selected university students can be asked to share their contacts with bright or motivated high school students who may want to learn more about Programming. The possibility of involving high school students in teams working on university projects or contests in the field of Computer Science should be explored.
Consultants in planning and designing the learning and teaching process	The selected university students should be fully engaged in the design of the hands-on activity in order to ensure that the tasks would be manageable for younger students without prior experience with programming. Students can be given the task to prepare the Power Point presentation for the activity, as well as any handouts and supporting materials. They should, however, do this on the basis of clear instructions from the faculty member who will lead the course.
Co-researchers contributing to subject-based	The selected university students can be asked

Co-funded by the Erasmus+ Programme of the European Union

research	to design small practical tasks intended to help students understand the syntax of Python. They should be instructed to keep the level of difficulty close to the skills and knowledge of high school students.

High School Teachers (supporting role is suitable for teachers in Computer Science or Physics)

Consultants in planning and designing the learning and teaching process	The accompanying teachers should have the leading role in selecting trainees from among the students. They should be approached in advance and consulted about the relevance of the presented examples and the level of difficulty of the theoretical presentation (in view of the intended group of trainees). Special attention should be paid to the selection of manageable practical tasks and the avoidance of tasks that have no relevance to the compulsory curriculum. Teachers should be consulted about the best way to draw parallels and to link the content of the course to the compulsory curriculum in Computer Science.
Pedagogical co-designers of learning, teaching and assessment; facilitators in hands-on and lab experiments	The accompanying teachers should work together with high school students during the practical activity in order to help them develop the extensions of the Python code, as well as to provide clarification to those of them who have failed to understand the presented material or apply it in practice. Most teachers would be in position in which they themselves will use Python for the first time, especially for developing a game. They should be given the chance to learn themselves. High School teachers should be the primary source of feedback about the effectiveness of the training. They will also be in the best position to assess the performance of their students. Teachers should play a central role in maintaining discipline during the activity.

University-high school partnerships

Co-funded by the Erasmus+ Programme of the European Union

This course in particular would be a suitable addition to study programs in secondary schools with a profile in Computer Science, Mathematics, or the Natural Sciences. It can be the beginning of a series of extra-curricular courses on Programming. If there is such an interest, contact between the accompanying teachers and the university should be made well in advance and the course should be planned as part of a larger set of topics. The course can be combined with workshops organized at the school by visiting university lecturers. One particular high school teacher or administrator and one particular university faculty member should be tasked with the organization of this Master Class and these people can later act as contact persons and "boundary spanners" for future collaboration. For further collaboration to be planned, it is advisable that an educational manager from the school attend (part of) the Master Class in order to witness the effectiveness of the training. If this is not possible, then a report on the achieved results and the satisfaction of students should be presented to the school management, together with a proposal for further collaboration.

Co-funded by the
Erasmus+ Programme
of the European Union