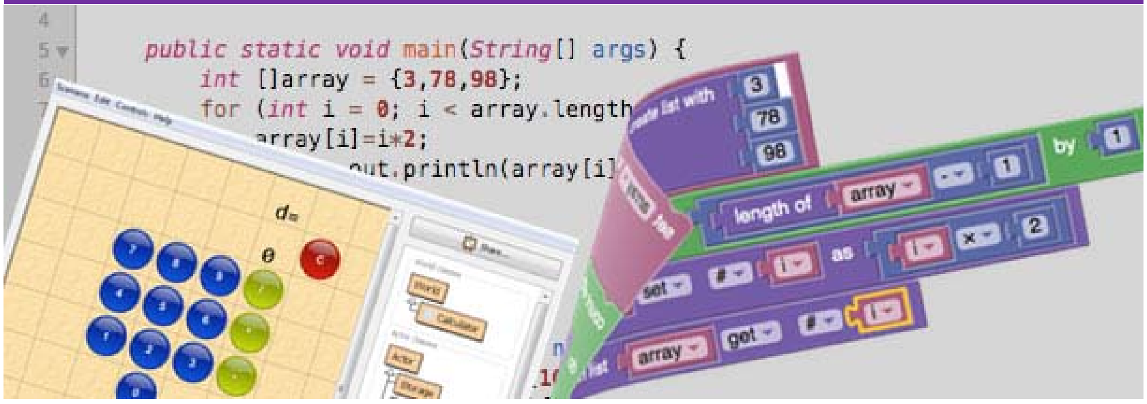
INTRODUCTION TO PROGRAMMING WITH JAVA



```
    public static void main(String[] args) {
        int []array = {3,78,98};
        for (int i = 0; i < array.length
            array[i]=i*2;
                out.println(array[i]
```

Part I:
Starting to program in Java

SYLLABUS

Universidad
Carlos III de Madrid

INTRODUCTION

Introduction to Programming with Java is an introductory course to learn programming in an easy and interactive way. This course is designed taking into account the subset and recommendations of the College Board in order to prepare learners for the Advanced Placement (AP) Computer Science A exam. The course is divided in three parts of five weeks each. This distribution is intended to make it easier for learners to work and practice at their own pace.

Part 1: Starting to Program in Java introduces programming on the basis of familiar concepts, like calculators and games. Powerful concepts such as functional abstraction, the object oriented programming (OOP) paradigm and Application Programming Interfaces (APIs) are progressively introduced throughout the five weeks. Examples and case studies are provided so that learners can implement simple programs on their own or collaborating with peers.

Learner's experience is enhanced through a wide range of videos and audiovisual materials. Exercises accompanying videos give immediate feedback on common pitfalls and misconceptions when programming in Java. Code is presented using three visual and simple tools that make the development of small programs easier: Blockly, Codeboard and Greenfoot.

It is important that you know that some activities and exercises in this course may not be compatible with mobile devices (such as those running on iOS or Android). We recommend that you use a computer to ensure the best learning experience.

If this is your first course on edX, do not hesitate to enroll in the Demo course to get to know the courseware: https://www.edx.org/course/demox-edx-demox-1.

OBJECTIVES

After finishing this course, the learner should:

- Have acquired basic knowledge on algorithms and Java programming
- Be able to develop programs with conditionals and loops
- Be able to design and implement recursive algorithms
- Know the basic mechanisms of the OOP paradigm
- Be able to use and interpret the API of some of the most common Java classes
- Be able to develop simple programs in Java

COURSE STAFF

- CARLOS DELGADO KLOOS: Full Professor at Universidad Carlos III de Madrid, Director of the UNESCO Chair on "Scalable Digital Education for All", and Vice-Rector for Infrastructures and Environment. He introduced the teaching of Java at Universidad Carlos III de Madrid in 1997. Content creator and general supervisor of the MOOC.
- CARMEN FERNÁNDEZ PANADERO: Assistant Professor at Universidad Carlos III de Madrid. Content creator and instructional designer.

- IRIA ESTÉVEZ-AYRES: Assistant Professor at Universidad Carlos III de Madrid. Content creator and assessment designer.
- JORGE BLASCO: Former Assistant Lecturer at Universidad Carlos III de Madrid and Research Fellow at City University London. Content creator and quality controller.
- CARLOS ALARIO-HOYOS: Teaching Assistant at Universidad Carlos III de Madrid. Content creator and instructional designer.
- SERGIO PASTRANA: Teaching Assistant at Universidad Carlos III de Madrid. Content creator and assessment designer.
- GUILLERMO SUAREZ-TANGIL: Teaching Assistant at Universidad Carlos III de Madrid. Content creator and responsible for communication with learners.
- JULIO VILLENA ROMÁN: Part-time lecturer at Universidad Carlos III de Madrid. Content creator and quality controller.

COURSE STRUCTURE

WEEK 1: *From the Calculator to the Computer*

The first week introduces basic programming concepts, such as values and expressions, as well as making decisions when implementing algorithms and developing programs.

WEEK 2: *State Transformation*

The second week introduces state transformation including representation of data and programs as well as conditional repetition.

WEEK 3: *Functional Abstraction*

The third week addresses the organization of code in a program through methods, which are invoked to carry out a task and return a result as answer. Recursion, as a powerful mechanism in the invocation of methods, is also covered this week.

WEEK 4: *Object Encapsulation*

The fourth week introduces the object oriented programming (OOP) paradigm, which enables the modeling of complex programs in Java through objects and classes. The concept of inheritance as the basis for reusing code and simplifying programs in Java is studied in this week.

WEEK 5: *Packaging*

The last week of the module aims to study the reuse of code through third-party classes that are already developed and that we can incorporate to our programs to perform specific actions, and reduce the number of lines that we need to code.

COURSE METHODOLOGY

Every week follows the same methodology and structure. First, theoretical concepts are presented in videos and other audiovisual formats in a simple and pleasant way through examples and metaphors. The learner assimilates these concepts by practicing with exercises, receiving immediate feedback.

Next, a case study is introduced to demonstrate the theoretical concepts. The learner can download, analyze and modify the code of the case study to improve the understanding on the concepts taught. At the end of the week learners can share their work with coursemates before taking the assessment activities. Supplementary materials to delve into the topics of the course may be provided.

The estimated time learners need to complete each week is from 5 to 7 hours.

COMMUNICATION AND SOCIAL COMPONENT

EMAILING and the **COURSE INFO PAGE** will be used by teachers to keep learners up-to-date with all the news related to the course.

In addition, **SOCIAL TOOLS** will be supported for learners to communicate with teachers and peers: the course forum on edX and Twitter (#javaedxuc3m). Programming can be very challenging, and difficulties will inevitably arise. Learners are encouraged to actively interact with other learners and teachers through these three social tools and share their concerns, problems, experiences and pieces of code.

EVALUATION

Evaluation will cover theoretical concepts and also small programs that learners will have to code. These activities are mandatory only for those who wish to get a certificate at the end of the course.

Theoretical concepts will be evaluated through weekly **EXAMS** (graded tests) and will have a weight of 75% in the final grade. The programming of code will have a weight of 25% in the final grade and will be based on two **PEER ASSESSMENTS (P2P)**, one in week 3 (10%) and the other one in week 5 (15%).

To **PASS THE COURSE** it will be necessary to obtain the 60% of the final grade.

CALENDAR

All the weeks will follow the same structure. Materials and self-evaluation activities (non graded) will be released on Tuesdays (12:00 UTC) and will be available until June 30 (23:59 UTC), last day of the course. The contents of week 2 will not be available until May 12, so that learners of different levels and with different background can easily catch up.

Please, pay attention to the calendar of exams (graded tests) and peer assessments (programming activities). Weekly exams will be available for two weeks, except the first one that will be opened for three weeks. Peer assessments include two weeks for the submission of your work and one additional week for reviewing the work of three fellows (see below); it is mandatory to submit your work in order to assess other people's work.

WEEK	RELEASE DATES (12:00 UTC)	CONTENTS
1	28 April	**From the Calculator to the Computer** • Exam 1 (graded test): due on 19 May (11:59 UTC)
2	12 May	**State Transformation** • Exam 2 (graded test): due on 26 May (11:59 UTC)
3	19 May	**Functional Abstraction** • Exam 3 (graded test): due on 2 June (11:59 UTC) • Peer assessment 1 (programming) o Submission: due on 2 June (11:59 UTC) o Review: due on 9 June (11:59 UTC)
4	26 May	**Object Encapsulation** • Exam 4 (graded test): due on 9 June (11:59 UTC)
5	2 June	**Packaging** • Exam 5 (graded test): due on 16 June (11:59 UTC) • Peer assessment 2 (programming) o Submission: due on 16 June (11:59 UTC) o Review: due on 23 June (11:59 UTC)

The course *Introduction to Programming with Java – Part 1: Starting to Program in Java* will finish on the 30th of June. Certificates will be ready after this date.