Coding in Action Lab I
Python-03

1 Instructions

- Only this page will serve as reference: do not trust rumors.
- Watch out! This document could potentially change up before submission.
- Make sure you have the appropriate permissions on your files and directories.
- You have to follow the submission procedures for all your exercises.
- Your exercises will be checked and graded by your fellow classmates.
- On top of that, your exercises will be checked and graded by a program called Moulinette.
- Moulinette is very meticulous and strict in its evaluation of your work. It is entirely automated and there is no way to negotiate with it. So if you want to avoid bad surprises, be as thorough as possible.
- Moulinette is not very open-minded.
- These exercises are carefully laid out by order of difficulty - from the easiest to the hardest. We suggest you to solve them in the given order.
- Using a forbidden function is considered cheating. Cheaters get -42, and this grade is non-negotiable.
- If your program contains a syntax error, you'll get 0.
- You cannot leave any additional file in your directory other than those specified in the subject.
- Got a question? Ask your peer on the right. Otherwise, try your peer on the left.
- Your reference guide is called Google / man / the Web / ....
- Examine the examples thoroughly. Your code must *exactly* reproduce the examples, for the specified test cases. Beware that the examples could very well call for details that are not explicitly mentioned in the problem statement.
- By Odin, by Thor ! Use your brain !!!

Exercise 00: `Point`

Create a program that contains the definition for a class named `Point` with attributes x and y. When executed, the program asks the user to insert two points and prints their distance.

Turn-in directory:	`ex00/`
Files to turn in:	`point.py`
Allowed functions:	`print, eval, float`

Examples:

```
42~ > python point.py
Insert the coordinates of the first point: 1.5,2
Insert the coordinates of the second point: 2,0
Their distance is: 2.0615528128088303
42~ >

42~ > python -c 'from point import Point; print(Point(1,1).x)'
1
42~ >
```

Exercise 01: `Circle`

Create a program that contains the definition for a class named `Circle` with attributes `center` and `radius`, where `center` is a `Point` object and `radius` is a real number. When executed, the program instantiates a `Circle` object with center at $(150, 100)$ and radius 75 and prints a description of this object using a `__str__` method.

Turn-in directory:	`ex01/`
Files to turn in:	`circle.py`
Allowed functions:	`print`

Examples:

```
42~ > python circle.py
Circle of center (150, 100) and radius 75
42~ >

42~ > python -c 'from circle import Circle; print(Circle((1,1),4))'
Circle of center (1, 1) and radius 4
42~ >
```

Exercise 02: `Circle (v2)`

By adding to `Circle` a method `contains`, create a program that takes as arguments the coordinates of a point, and checks whether the points lies in or on the boundary of the circle of center $(0, 0)$ and radius 1.

Turn-in directory:	`ex02/`
Files to turn in:	`circle.py`
Allowed functions:	`print`

Examples:

```
42~ > python circle.py 1.5 1.5
The Point (1.5, 1.5) lies out of the Circle of center (0, 0) and radius 1
42~ >

42~ > python circle.py 0.5 0.4
The Point (0.5, 0.4) lies in the Circle of center (0, 0) and radius 1
42~ >

42~ > python -c 'from circle import Circle,Point; print(Circle((0,0),2).contains(Point(1,1)))'
True
42~ >
```

Exercise 03: `student`

Create a program that contains the definition of two classes, `Person` and `Student`, where `Student` "extends" `Person` "overriding" its `__str__` method. When executed, the program asks the user to insert their first and last name, and whether the user is a student. If the user is a student, the program also asks to which degree course the user is registered. Finally, the program prints a description of the user.

Turn-in directory:	`ex03/`
Files to turn in:	`student.py`
Allowed functions:	`print, input`

Examples:

```
42~ > python student.py
Insert first name: Stefano
Insert last name: Guarino
Are you a student? (y/n)y
Please insert your degree course: BA
Stefano Guarino is registered to BA
42~ >

42~ > python student.py
Insert first name: Stefano
Insert last name: Guarino
Are you a student? (y/n)n
Stefano Guarino
42~ >

42~ > python student.py
Insert first name: Stefano
Insert last name: Guarino
Are you a student? (y/n)yes
Please type "y" or "n": y
Please insert your degree course: CS
Stefano Guarino is registered to CS
42~ >

42~ > python -c 'from student import Student; print(Student("Stefano","Guarino"))'
Stefano Guarino is not registered to any course
42~ >
```

Submission and peer-evaluation

Turn in your assignment in your Git repository as usual. Only the work inside your repository will be evaluated during the defense. Do not hesitate to double check the names of your files to ensure they are correct.