

**i**   **Instructions**

This exam is made of two parts:

- Part 1 consists of 20 multiple choice questions. Most of them can be answered either by looking up [the Python documentation](#) or by trying out some code in a Python interpreter on your computer.
- In part 2, you will have to will have to write your own code according to some specifications.

To get grade 3, you need to get at least 10 correct answers in the part 1. For a higher grade, you also need at least 10 correct answers in part 1, and you need to answer all questions in part 2. The quality of your answers in terms of code quality and how well your solution works will then determine whether you get grade 3, 4, or 5. If you get less than 10 correct answers in part 1, part 2 will not be graded.

Make sure to structure your code properly, name your variables sensibly, make use of functions and loops whenever possible, and use appropriate data structures. Please note that unreadable snippets, unnecessary repetitions (e.g. copy/paste), uninformative variable names and badly ineffective code will affect your grade negatively.


Remember that you are allowed to try out code on your computer and to look up the Python documentation, the course materials, and your own code from the previous assignments. This exam is individual.

Igor and Adrien will be available throughout the exam in case you have any questions (igor.tominec@it.uu.se and adrien.coulier@it.uu.se).

**1.1**   Let's say we have a diction `d = {'Bob': [0, 1], 'Alice': [3, 9, 2], ...}`, where the keys are students' names, and the values are lists of grades they got for the assignments of this semester.

How can we create a list containing the students' names sorted by the number of assignments they have submitted this semester?

**Select one alternative:**

- ☐ `sorted(d.keys(), key=lambda k: d[k][0])`
- ☐ `sorted(d.keys())`
- ☒ `sorted(d.keys(), key=lambda k: len(d[k]))` 
- ☐ `sorted(d.keys(), key=lambda k: len(k))`

---

Maximum marks: 1

1.2 Given the following code and matplotlib's [documentation](#), what is the command to set the label on the x axis?

```
import matplotlib.pyplot as plt
plt.plot([1,2,3])
```

Select one alternative:

- ☐ ax.set\_xlabel("Days")
- ☒ plt.xlabel("Days")
- ☐ plt.set\_xlabel("Days")
- ☐ ax.xlabel("Days")



Maximum marks: 1

1.3 Given the following code and matplotlib's [documentation](#), what is the command to set the label on the y axis?

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()
ax.plot([1,2,3])
```

Select one alternative:

- ☒ ax.set\_ylabel("Days")
- ☐ plt.set\_ylabel("Days")
- ☐ plt.y\_label("Days")
- ☐ ax.y\_label("Days")



Maximum marks: 1

1.4 Given some text in a file, what is the type of the variable `text` in the following code?

```
f = open('text.txt', 'r')
text = f.read()
```

Select one alternative:

- ☒ str
- ☐ float
- ☐ list
- ☐ int



Maximum marks: 1

1.5 What is the first argument of a class method usually called?  
**Select one alternative:**

- ☐ `def`
- ☐ `super`
- ☐ `class`
- ☐ `self`



Maximum marks: 1

1.6 Given some text in a file, what is the type of the variable `text` in the following code?

```
f = open('text.txt', 'r')
text = f.readlines()
```

**Select one alternative:**

- ☐ float
- ☐ list
- ☐ int
- ☐ str



Maximum marks: 1

1.7 What is the name of the method called to create an instance of a class?  
**Select one alternative:**

- ☐ `\_\_construct\_\_`
- ☐ `\_\_init\_\_`
- ☐ `\_\_build\_\_`
- ☐ `\_\_create\_\_`



Maximum marks: 1

1.8 Following the [documentation of the math module](#), which function, given a floating point number `x`, returns the largest integer smaller or equal to `x`?

Select one alternative:

- ☐ round
- ☒ floor
- ☐ max
- ☐ ceil



Maximum marks: 1

1.9 In the following loop, what is the type of variable `i`:

```
contains_o = False
word_list = ['This', 'is', 'a', 'list', 'of', 'words']
for i in range(len(word_list)):
    if 'o' in word_list[i]:
        contains_o = True
```

Select one alternative:

- ☐ a list
- ☒ an integer
- ☐ a word
- ☐ a string



Maximum marks: 1

1.10 I want to print the content of a file, as well as how many lines it has, but my code does not work, it always prints that my file has zero lines.

Why doesn't my code count the number of lines properly?

```
filename = 'data.txt'
f = open(filename, 'r')
lines = f.read()
print(lines)
n_lines = len(f.readlines())
print(f"The file {filename} has {n_lines} lines.")
```

Select one alternative:

- ☐ Files should always be opened with the `from` keyword
- ☒ A file can only be read one single time after it has been opened.
- ☐ The file should be closed at the end of the program.
- ☐ The file should be read with the `read` method the second time.



Maximum marks: 1

1.11 Following the [documentation of the math module](#), which function, given a floating point number `x`, returns the truncated value of this number (i.e. without the decimals)?

Select one alternative:

- ☐ ceil
- ☒ trunc
- ☐ round
- ☐ truncate



Maximum marks: 1

1.12 What would be the best suitable name for a class describing a class of students, according to the [PEP8](#) convention?

Select one alternative:

- ☐ str
- ☐ class
- ☐ student\_class
- ☒ StudentClass



Maximum marks: 1

1.13 I wrote the following code to print all numbers from 0 to n in ascending order, but it never stops, what is the problem?

```
n = 10
i = 0
i += 1
while i <= n:
    print(i)
```

Select one alternative:

- ☐ the condition should be `i > n`
- ☒ `i += 1` should be inside the loop
- ☐ `n` should be 11
- ☐ `i` should start at 1



Maximum marks: 1

1.14 In the following code, what is `n`?

```
def f():  
    return 3  
n = f()
```

Select one alternative:

- ☐ a list
- ☒ an integer
- ☐ a tuple
- ☐ a function



Maximum marks: 1

1.15 In the following loop, what is the type of the variable `word`:

```
contains_o = False  
word_list = ['This', 'is', 'a', 'list', 'of', 'words']  
for word in word_list:  
    if 'o' in word:  
        contains_o = True
```

Select one alternative:

- ☐ a list
- ☐ a word
- ☐ an integer
- ☒ a string



Maximum marks: 1

1.16 Given, the class below, which methods are being called in the example:

```
class City:
    def __init__(self, name='', pop=0):
        self.name = name
        self.pop = pop
    def __str__(self):
        info = f'The city {self.name} has {self.pop} population.'
        return info
    def add_pop(self, n):
        self.pop += n
city = City('Uppsala', 233295)
print(city)
```

Select one or more alternatives:

- ☒ \_\_str\_\_ ✓
- ☒ \_\_init\_\_ ✓
- ☐ add\_pop
- ☐ City

Maximum marks: 1

1.17 What would be the best suitable name for a variable containing a String representing the name of a student, following the [PEP8](#) conventions?

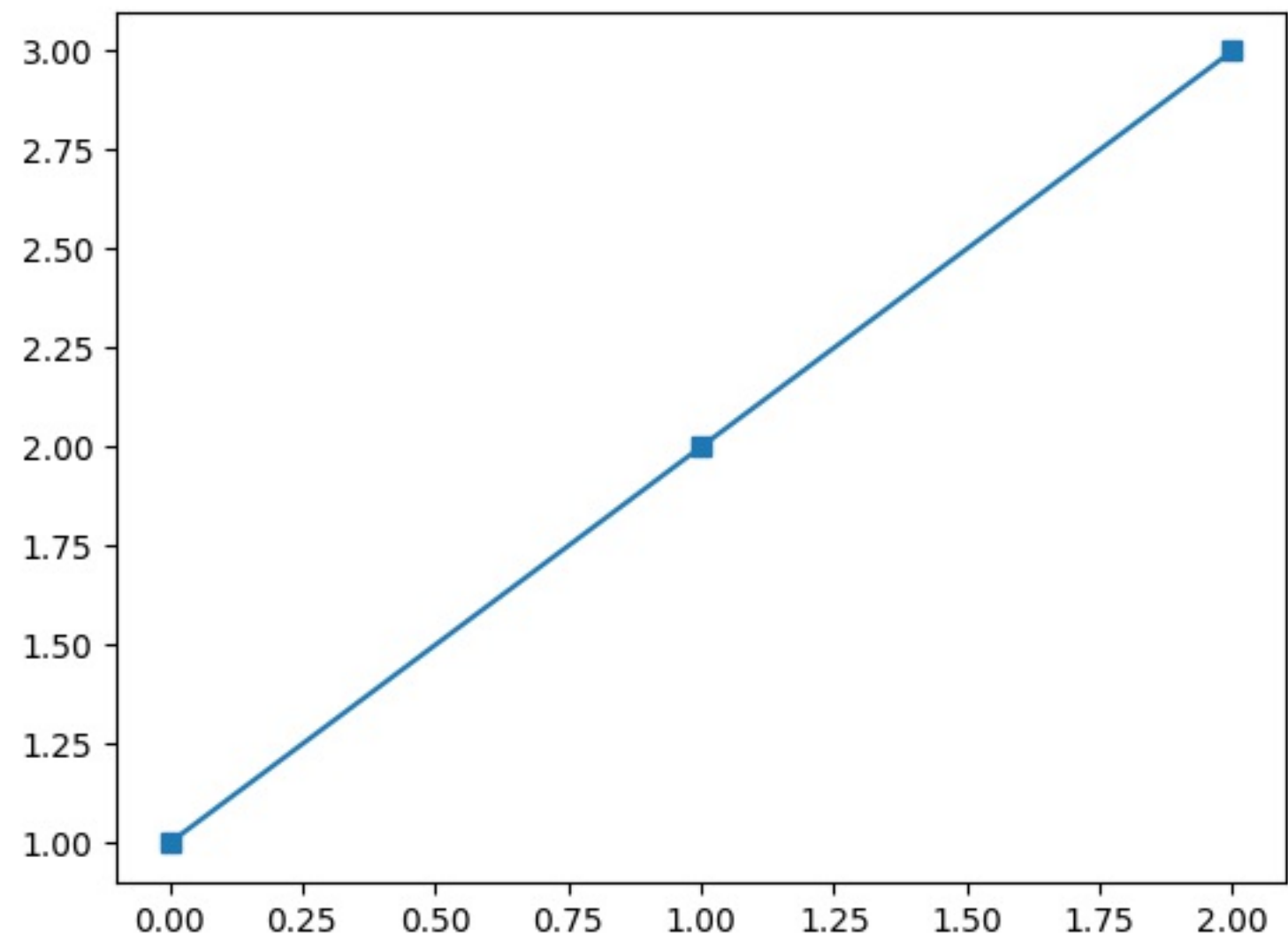
Select one alternative:

- ☐ n
- ☐ StudentName
- ☒ student\_name ✓
- ☐ str

Maximum marks: 1

1.18 Following [matplotlib's documentation](#), what should be the value of variable `mrk` so that the data points are drawn as squares, like in the plot below?

```
plt.plot([1,2,3], marker=mrk)
```



Select one alternative:

- ☐ "c"
- ☐ "square"
- ☒ "s"
- ☐ "sqr"



Maximum marks: 1

1.19 According to the [string documentation](#), how can one remove whitespaces at the beginning of a string (that is, ' text' becomes 'text')?

Select one alternative:

- ☐ There is no such function
- ☒ lstrip
- ☐ remove\_whitespace
- ☐ trim



Maximum marks: 1



**1.20** Python’s style conventions are described by [PEP8](#). Which of the following lines follows PEP8?  
**Select one alternative:**

- ☐ `import sys, os`
- ☒ `dct['key'] = lst[index]`
- ☐ `spam (1)`
- ☐ `spam( ham[ 1 ], { eggs: 2 } )`



Maximum marks: 1

**1.21** After the following code is executed, what is the value of variable `x`:

```
text = "Python is an interpreted, high-level and general-purpose programming language."  
x = text[1:6]
```

**Select one alternative:**

- ☐ an integer
- ☐ a dict
- ☒ a string
- ☐ a list



Maximum marks: 1

**1.22** Let's say we have a diction `d = {'Bob': [0, 1], 'Alice': [3, 9, 2], ...}`, where the keys are students' names, and the values are lists of grades they got for the assignments of this semester.

How can we create a list containing the students' names sorted by the grade they got for the first assignment they submitted?

**Select one alternative:**

- ☐ `sorted(d.keys())`
- ☒ `sorted(d.keys(), key=lambda k: d[k][0])`
- ☐ `sorted(d.keys(), key=lambda k: len(k))`
- ☐ `sorted(d.keys(), key=lambda k: len(d[k]))`



Maximum marks: 1

1.23 After the following code is executed, what is the type of variable `x`

```
text = "Python is an interpreted, high-level and general-purpose programming language."
words = text.split()
x = words[1:6]
```

Select one alternative:

- ☐ a string
- ☐ a dict
- ☐ an integer
- ☐ a list



Maximum marks: 1

1.24 I wrote the following code to print all numbers from n to 0 in descending order, but it never stops, what is the problem?

```
n = 10
i = n
while i <= n:
    print(i)
    i -= 1
```

Select one alternative:

- ☐ It should be `i += 1` instead of `i -=1`
- ☐ `i -= 1` should be outside the loop
- ☐ The condition should be `i >= 0`
- ☐ `i` should be initialized to 0



Maximum marks: 1

1.25 I have written the following code, but I was told that `f` is not a very sensible name for that function. What would be the best suitable name instead?

```
def f(wordlist):
    count = 0
    for word in wordlist:
        if len(word) > 5:
            count += 1
    return count
```

Select one alternative:

- ☐ count
- ☐ words
- ☐ count\_long\_words
- ☐ count\_short\_words



Maximum marks: 1

**1.26** Given a list `lst`, what does `lst[::-2]` return?  
**Select one alternative:**

- ☒ Every other element of `lst`.
- ☐ The second to last value in `lst`.
- ☐ All the elements in the list, except the first one.
- ☐ Nothing, it does not work.



Maximum marks: 1

**1.27** In the following code, what is `g`?

```
def f():  
    return 3  
g = f
```

**Select one alternative:**

- ☒ a function
- ☐ an integer
- ☐ a tuple
- ☐ a list



Maximum marks: 1

## i Problem Description

For these questions, we would like to study how child mortality relates to the number of children per woman, all around the world and over the last decades. The UN and the World Banks are great, reliable resources to get such data. [The Gapminder Foundation](#) is also a great resource, that compiles data sets from different sources and makes them available for everybody to see, through all kind of visualization tools.

Following the links below, you will find data (compiled by the Gapminder Foundation) regarding child mortality, babies per woman, population and continent, for most countries in the world, from 1800 to the present day, as well as how these values are forecast to evolve in the future, according to the UN.

[child\\_mortality](#)  
[total\\_fertility\\_rate](#)  
[population](#)  
[country-continent](#)

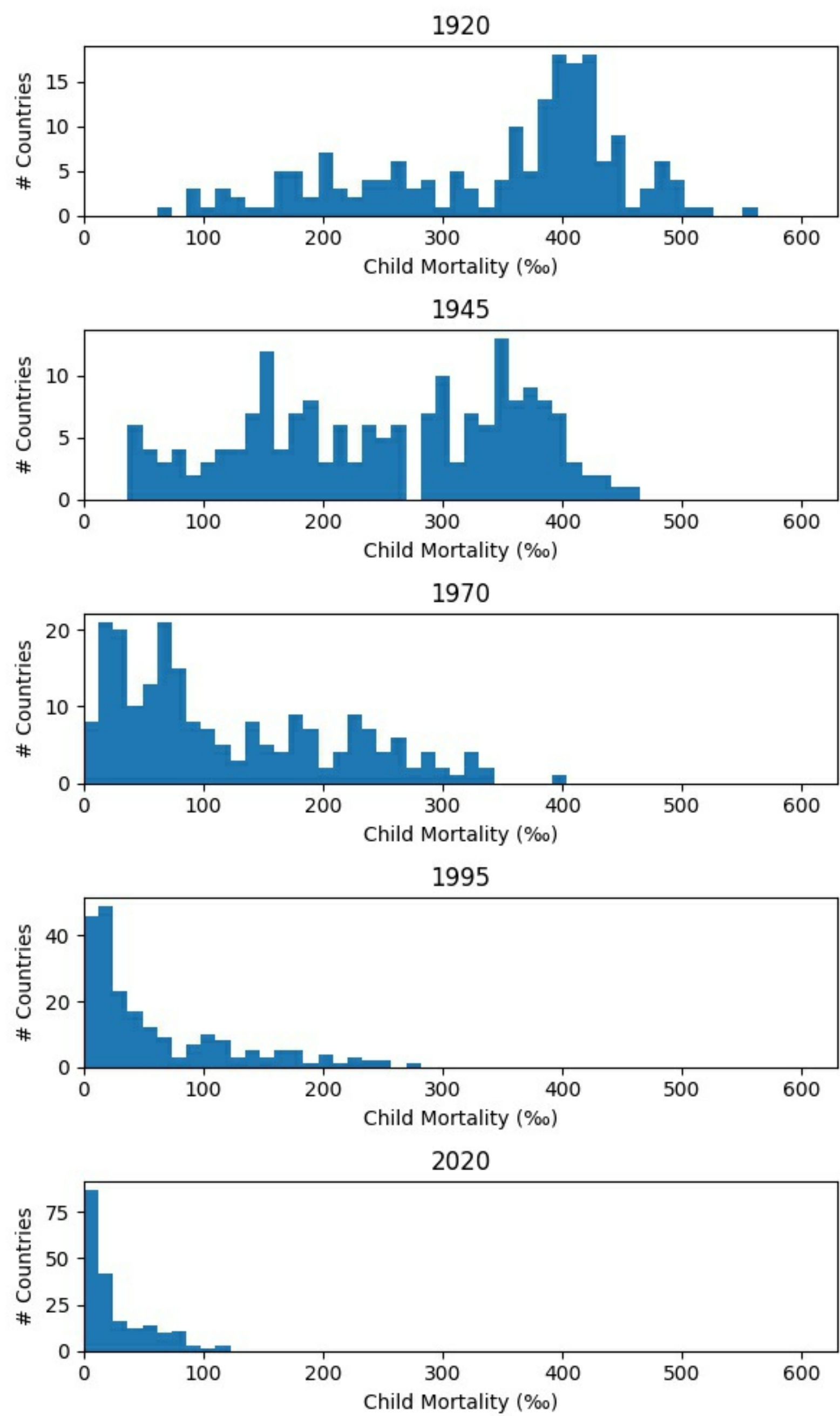
NB: as is often the case with real datasets, the format is not quite the same for all files, sometimes some numbers are missing, and some files might be missing some countries as well. Remember to make your code as robust as possible to these discrepancies.

In the next two questions, you will be invited to reproduce some plots, using matplotlib and the datasets provided here. As much as possible, use loops and functions to avoid repeating the same lines of code more than necessary. Your plots should look as similar as possible to the ones provided, and you are encouraged to use the matplotlib documentation or any other resources available online. Major differences between your plots and our plots will affect your grade negatively, as well as bad code quality (problems with variable names, legibility, conciseness or efficiency).

If you decide to use code from examples found online (e.g. on StackOverflow), we ask you to provide a link (e.g. as a comment in your code) to the source you used and to describe what you think this snippet does, in your own words. Failure to follow these guidelines will negatively affect your grade.

- 2.1** First, we would like to get an idea of how child mortality has evolved throughout the world, over the last 100 years. Specifically, we would like to plot the distribution of child mortality for years 1920, 1945, 1970, 1995 and 2020. For instance, in the plot below, we can see that back in 1920, the mortality rate was about 400 deaths per 1000 births for most countries in the world. At the present day, in 2020, we can see that this rate has dropped below 25‰ for almost every country! In fact, the worst countries today are better than the best countries one hundred years ago.

Using the data provided in the problem description, write a program that produces the histograms below, using matplotlib. Remember that you can use Matplotlib's documentation. For instance, the documentation of the histogram function can be found [here](#). Do make sure that the x scale and the bins are the same for all plots so that they are comparable and make sure all the plots are part of the same figure (i.e. do use `plt.subplots`).



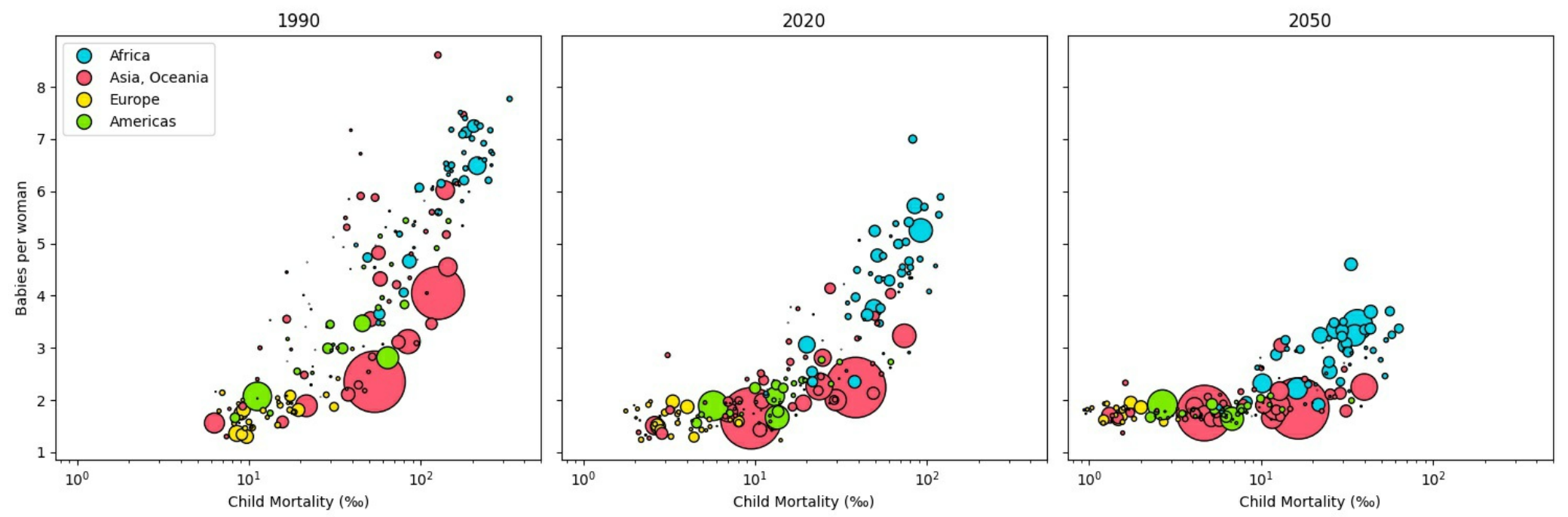
1	
---	--

Maximum marks: 10

**2.2** We now want to look at how child mortality relates to babies per woman, population and continent, for years 1990, 2020, and for the UN forecast for 2050. Do do this we will use a scatter plot, where the x-axis will represent child mortality, the y axis will represent babies per woman, the size of each dot will be related to the population, and countries will be grouped by color depending on the continent they belong to.

For instance, in the plot below, we can see that most countries have already transitioned to a low population growth state, and that all countries are expected to do so in the near future, according to UN experts.

Using the data provided in the instructions, write a program that produces the following scatter plots. Again, put all the plots in the same figure (using `plt.subplots``), and make sure the axes are scaled properly. Minor differences (e.g. small differences with the dot sizes for instance) will be tolerated.



Fill in your answer here

1

Maximum marks: 10

**i** **Problem Description**

[Conway's Game of Life](#) is a cellular automaton, where each cell on a grid may be alive or dead depending on the number of living cells in the 8 neighboring grid points, according to the following rules:

- 1. Any live cell with two or three live neighbors survives.
- 2. Any dead cell with three live neighbors becomes a live cell.
- 3. All other live cells die in the next generation. Similarly, all other dead cells stay dead.

The figure bellows shows all 8 neighbors considered for a given cell:



123  
4#5  
678

The grid is updated by step: at each time step, all the cells are updated *simultaneously* depending on how many neighbors they have in the current step. For instance, in the example below, the cell on the top side dies after the first step because it only has one neighbor. The cell just below survives, because it has two neighbors (although these two neighbors are going to die in the next step).

For this exercise, we would like to implement a small class to simulate such automaton.

This class will have a single attribute (a list of lists of boolean values, True or False) to represent the grid (a grid point will have the value True if there is a live cell there and False otherwise).

Again, bad code quality will affect your grade negatively. Questions 2b and 2c will only be graded if you provide an implementation of the Grid class in question 2a.

Below, you'll find an example running a small grid for 4 timesteps, printing the grid at every timestep.

```
..#.
###.
..#.

.#..
..##
.#..

..#.
.##.
..#.

.##.
.###
.##.

.#.#
#..#
.#.#
```



2.3 Implement a class **Grid** according to the specification from the problem description.

Your class should include the following methods:

- An **\_\_init\_\_(self, grid)** function that takes a list of lists of boolean values as unique parameter and stores it as an attribute. For convenience, it might be practical to create attributes to store the size of this grid as well.
- **flip\_cell(self, coord)**, that flips over the cell situated at the given coordinates, i.e. if the cell is currently dead it becomes alive and if it is already alive it becomes dead.
- **n\_neighbors(self, coord)** that takes some coordinates and returns the number of living neighbors surrounding the cell at that position. Note that cells situated on the edge of the grid will have fewer neighbors to consider.
- **step(self)** that computes the next state of the grid. For this method, since all cells must be updated simultaneously, you will need to make a copy of the old grid before you compute the new one, otherwise, cells might use neighbors from the new time step to update themselves.
- **print\_snapshot(self)** that prints the current state of the grid, with a dot '.' if the grid point is empty, and a '#' symbol if there is a living cell.
- **update(self, n, snap\_freq)** that updates the grid by n steps, and prints a snapshot every 'snap\_freq' step.

Make sure to make your code is legible and add docstrings describing each method whenever necessary. Remember that you are allowed to test the code on your own computer and to look up the documentation.

Finally, please make sure you try out your code on your own device before you submit it.

Fill in your answer here

1

Maximum marks: 10

**2.4** Let's say we would like to make the neighboring rules more flexible, that is, we want to be able to provide custom rules to the constructor and use that to update the grid. For instance, maybe we would like to simulate a system where living cells with 2, 5 or 7 neighbors survive and dead cell with 3, 4 or 5 neighbors becomes alive, or vice versa, or any set of values really. What would be a good data structure to represent these rules? What would be the new signature of the `__init__` function? Explain how this data structure would be used and motivate your choice below.

**Fill in your answer here**

Maximum marks: 5

