Assignment 1

COMP9021, Trimester 1, 2023

1. General matters

1.1. **Aim.** The purpose of the assignment is to:

- develop your problem solving skills;
- design and implement the solutions to problems in the form of small sized Python programs;
- practice the use of arithmetic computations, tests, repetitions, lists, dictionaries, strings.

1.2. **Submission.** Your programs will be stored in files named `poker_dice.py` and `perimeter.py`. After you have developed and tested your program, upload it using Ed (unless you worked directly in Ed). Assignments can be submitted more than once; the last version is marked. Your assignment is due by March 27, 10:00am.

1.3. **Assessment.** The assignment is worth 13 marks. It is going to be tested against a number of inputs. For each test, the automarking script will let your program run for 30 seconds.

Late assignments will be penalised: the mark for a late submission will be the minimum of the awarded mark and 13 minus the number of full and partial days that have elapsed from the due date.

The outputs of your programs should be **exactly** as indicated.

1.4. **Reminder on plagiarism policy.** You are permitted, indeed encouraged, to discuss ways to solve the assignment with other people. Such discussions must be in terms of algorithms, not code. But you must implement the solution on your own. Submissions are routinely scanned for similarities that occur when students copy and modify other people's work, or work very closely together on a single implementation. Severe penalties apply.

2. Poker dice (7 marks)

Write a program named `poker_dice.py` that simulates the roll of 5 dice, at most three times, as described at

http://en.wikipedia.org/wiki/Poker_dice

as well as a given number of rolls of the 5 dice to evaluate the probabilities of the various hands.

The next three pages show a possible interaction.

The following observations are in order.

- Your program has to define the functions `play()` and `simulate()`, but of course it will likely include other functions.
- What is input from the keyboard is displayed in red; what is black is output by the program (in the pdf, for clarity, not at the prompt…).
- The hands are displayed in the order Ace, King, Queen, Jack, 10 and 9.
- All dice can be kept by inputting either `all`, or `All`, or the current hand in any order.
- No die is kept by inputting nothing.
- We have control over what is randomly generated by using the `seed()` function at the prompt, but do not make use of `seed()` in the program.
- To roll a die, we use `randint(0, 5)` **with 0, 1, 2, 3, 4 and 5 corresponding to Ace, King, Queen, Jack, 10 and 9, respectively**.

:

```
$ python3
...
>>> from random import seed
>>> import poker_dice
>>> seed(0)
>>> poker_dice.play()
The roll is: Ace Queen Jack Jack 10
It is a One pair
Which dice do you want to keep for the second roll? all
Ok, done.
>>> poker_dice.play()
The roll is: Queen Queen Jack Jack Jack
It is a Full house
Which dice do you want to keep for the second roll?  All
Ok, done.
>>> poker_dice.play()
The roll is: King King Queen 10 10
It is a Two pair
Which dice do you want to keep for the second roll?  King 10 Queen King 10
Ok, done.
>>> poker_dice.play()
The roll is: Ace King Queen 10 10
It is a One pair
Which dice do you want to keep for the second roll? 10 11
That is not possible, try again!
Which dice do you want to keep for the second roll? ace
That is not possible, try again!
Which dice do you want to keep for the second roll? 10 10
The roll is: King 10 10 10 9
It is a Three of a kind
Which dice do you want to keep for the third roll? all
Ok, done.
>>> poker_dice.play()
```

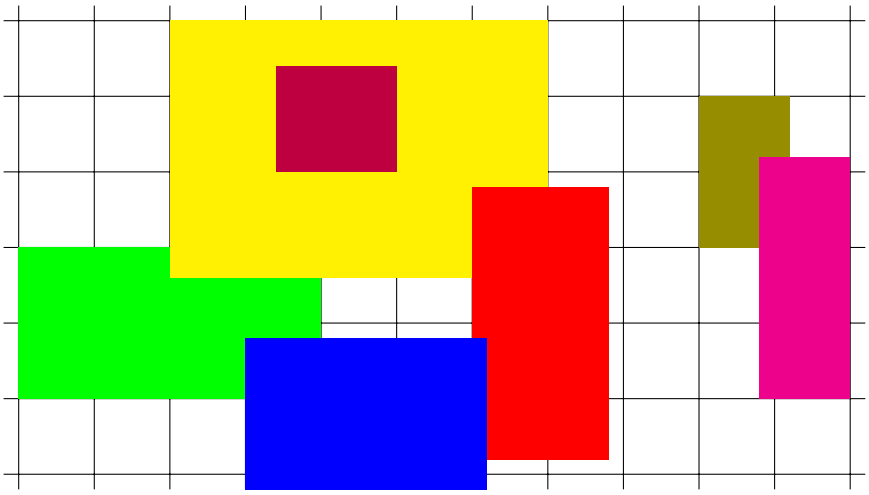
```
The roll is: Ace Ace Queen 9 9
It is a Two pair
Which dice do you want to keep for the second roll? Ace
The roll is: Ace Ace Queen Jack 10
It is a One pair
Which dice do you want to keep for the third roll? Ace
The roll is: Ace Queen Queen Jack 10
It is a One pair
>>> seed(2)
>>> poker_dice.play()
The roll is: Ace Ace Ace King Queen
It is a Three of a kind
Which dice do you want to keep for the second roll? Ace Ace Ace
The roll is: Ace Ace Ace 9 9
It is a Full house
Which dice do you want to keep for the third roll? all
Ok, done.
>>> poker_dice.play()
The roll is: King Queen Queen 10 10
It is a Two pair
Which dice do you want to keep for the second roll?
The roll is: Ace King Jack 10 9
It is a Bust
Which dice do you want to keep for the third roll?
The roll is: Queen Jack 10 9 9
It is a One pair
>>> seed(10)
>>> poker_dice.play()
The roll is: Ace Jack Jack 10 10
It is a Two pair
Which dice do you want to keep for the second roll? Jack 10 Jack 10
The roll is: Ace Jack Jack 10 10
It is a Two pair
Which dice do you want to keep for the third roll? Jack 10 Jack 10
The roll is: King Jack Jack 10 10
It is a Two pair
>>> seed(20)
>>> poker_dice.play()
The roll is: King Queen 9 9 9
It is a Three of a kind
Which dice do you want to keep for the second roll? 9 King 9 9
The roll is: King 9 9 9 9
It is a Four of a kind
Which dice do you want to keep for the third roll? 9 9 9 9
The roll is: Ace 9 9 9 9
It is a Four of a kind
```

```
>>> seed(0)
>>> poker_dice.simulate(10)
Five of a kind : 0.00%
Four of a kind : 0.00%
Full house     : 10.00%
Straight       : 0.00%
Three of a kind: 0.00%
Two pair       : 20.00%
One pair       : 60.00%
>>> poker_dice.simulate(100)
Five of a kind : 0.00%
Four of a kind : 0.00%
Full house     : 3.00%
Straight       : 4.00%
Three of a kind: 14.00%
Two pair       : 28.00%
One pair       : 45.00%
>>> poker_dice.simulate(1000)
Five of a kind : 0.10%
Four of a kind : 2.50%
Full house     : 3.80%
Straight       : 3.40%
Three of a kind: 17.20%
Two pair       : 20.60%
One pair       : 46.30%
>>> poker_dice.simulate(10000)
Five of a kind : 0.08%
Four of a kind : 1.99%
Full house     : 3.93%
Straight       : 3.33%
Three of a kind: 14.88%
Two pair       : 23.02%
One pair       : 46.58%
>>> poker_dice.simulate(100000)
Five of a kind : 0.08%
Four of a kind : 1.94%
Full house     : 3.85%
Straight       : 3.17%
Three of a kind: 15.47%
Two pair       : 23.02%
One pair       : 46.31%
```

3. Overlapping photographs (6 marks)

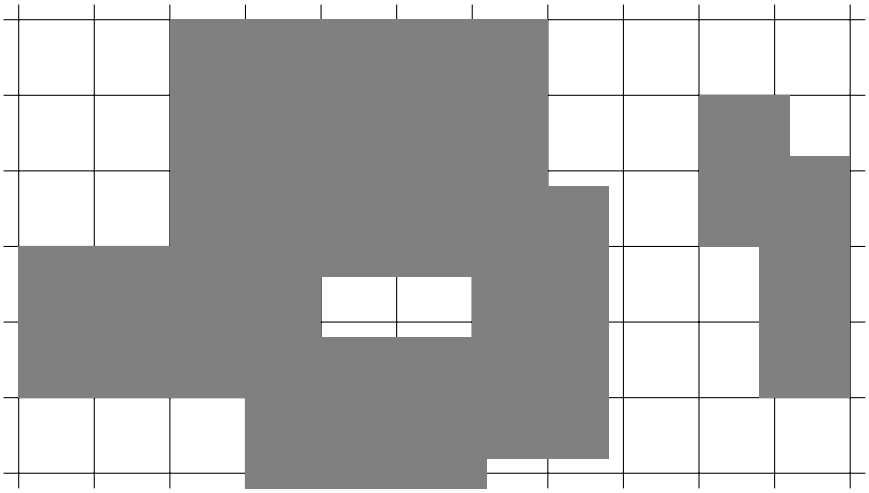
Write a program, stored in a file named `perimeter.py`, that performs the following task.

- Prompts the user to input the name of text file assumed to be stored in the working directory. We assume that if the name is valid then the file consists of lines all containing 4 integers separated by whitespace, of the form x_1 y_1 x_2 y_2 where (x_1, y_1) and (x_2, y_2) are meant to represent the coordinates of two opposite corners of a rectangle. With the provided file `frames_1.txt`, the rectangles can be represented as follows, using from top bottom the colours green, yellow, red, blue, purple, olive, and magenta.



We assume that all rectangles are distinct and either properly overlap or are disjoint (they do not touch each other by some of their sides or some of their corners).

- Computes and outputs the perimeter of the boundary, so with the sample file `frames_1.txt`, the sum of the lengths of the (external or internal) sides of the following picture.



Here is a possible interaction with the two provided sample files.

```
$ python3 perimeter.py
Which data file do you want to use? frames_1.txt
The perimeter is: 228
$ python3 perimeter.py
Which data file do you want to use? frames_2.txt
The perimeter is: 9090
```