CPSC 217 Assignment 1

Due: Wednesday May 22, 2024

- Part 1 is due at 11:55pm
- Part 2 is due at the end of class (4:30pm)
- Part 3 is due at the end of class (4:30pm)

Weight: 6%

Individual Work:

All assignments in this course are to be completed individually. Use of large language models, such as ChatGPT, and/or other generative AI systems is prohibited. Students are advised to read the guidelines for avoiding plagiarism located on the course website. Students are also advised that electronic tools may be used to detect plagiarism.

Late Penalty:

Late assignments will not be accepted.

Submission Instructions:

This assignment includes three parts. The first part requires you to write a computer program using Python. Your program must be submitted electronically. Use the Assignment 1 drop box in D2L for the electronic submission. No paper submission is required for the first part of the assignment.

The second part of this assignment consists of written questions that you will complete and submit on paper. They can be hand written (as long as they are legible) or typed. Submit the second part of the assignment to your TA during your tutorial time or to Ben during class time. There is no electronic submission for the second part of the assignment.

The third part of the assignment consists of one Parson's Problem that you will complete and submit on a bubble sheet. A bubble sheet can be picked up in class, or you can print one from the course website (as long as you use a good quality printer). Submit this part of the assignment to Ben during class, office hours, or by sliding it under Ben's office door (ICT 704). Note that the 7th floor to the ICT building locks at 5:00pm on business days and is locked on the weekend.

Posting Submissions for Public Viewing:

In previous years, we have received many creative and highly artistic submissions for the first part of this assignment. As such, we plan to post the images that are created on the course webpage so that others can view them. Your image will be posted anonymously, unless you choose to include your name as part of the image that you create. (Do **not** include your student number on the image). If you are **not** willing to have your image included on the website then please send an email to ben.stephenson@ucalgary.ca clearly stating such.

Part 1: Creating a Graphical Python Program

In this part of the assignment, you will create a small graphical Python program. The program will read two integers as input from the user. Its output will be a "smiley" (loosely defined) centered on the position specified by the user. For example, if the user enters 400 and 300 as input then the face should be **centered** in the window. If the user enters 0 and 0 then the face should be **centered** in the upper left corner of the window, with part of it cut off by the edge of the window. Use the SimpleGraphics library that you were introduced to previously to complete this task.

Getting Input:

Read your input using the techniques that we have discussed in class. Display a prompt in the terminal window with a print statement, or by providing a parameter to the input function. Note that the user will enter their input in the terminal window, not in the graphics window.

Recommended Algorithm:

There are many ways to create the program described in this section of the assignment. If you are having trouble getting started, you may want to consider creating your solution by using the following steps:

1. Import the SimpleGraphics library
2. Prompt the user to enter the x-position by displaying a message in the terminal
3. Read the x position from the user as a number
4. Prompt the user to enter the y-position by displaying a message in the terminal
5. Read the y position from the user as a number
6. Draw the face by calling functions in the SimpleGraphics library. These function calls will likely include calculations involving x and y

Additional Specifications:

Ensure that your program meets all of the following requirements:

- The image generated by your program should be some sort of a face that uses most of the drawing area.
- The image generated by your program must use at least 4 distinct colors.
- The image generated by your program must use at least 4 different graphics primitives such as ellipses, polygons, rectangles, text, etc.
- Your program may not load images or use other features that require us to have files beyond your .py file. Please use standard fonts such as Arial or Times if you choose to include text in your image.
- Do not resize the window.
- Your program should read only two input values – the x-position and the y-position of the center of the face.
- Your program must include appropriate comments, including a comment at the top of your file which includes **your name and student number** and describes the purpose of your program. There should also be comments within the program that indicate which lines of code draw different parts of the face (eyes, nose, hair, ears, mouth, etc.).
- You do **not** need to replace numbers used to control the positions of shapes with named constants.
- Do **not** display your student number as part of the image.

Part 2: Information and Data

Solve the following problems, and submit your answers on paper. Make sure you include your name and ID number on your submission. It is not necessary to type your answers as long as your work is reasonably neat and easy to read. An electronic submission is **not** required for this part of the assignment.

1: [4 marks] Convert the following base 10 numbers to binary:

a) 86
b) 763
c) 3667
d) 27538

2: [4 marks] Repeat question 1, converting each base 10 value to base 8:

3: [4 marks] Repeat question 1, converting each base 10 value to hexadecimal:

4: [4 marks] Repeat question 1, converting each base 10 value to base 5:

5: [4 marks] Convert the following base 2 numbers to decimal:

a) 10111
b) 111010
c) 1010110
d) 10101101111

6: [4 marks] Convert the following numbers:

a) 3232 base 4 to base 10
b) 2754 base 8 to base 10
c) 100 base 16 to base 10
d) 100 base 7 to base 10

7: [8 marks] Convert the following numbers:

a) E2 base 16 to base 6
b) 1111001111 base 2 to base 3
c) BAD base 14 to base 20
d) 5345 base 6 to base 16

8: [12 marks] Answer the following questions. Your response to each question should be brief (three sentences or less).

a) In ASCII, what value represents the letter A?
b) Is the value assigned to the letter A arbitrary, or was it selected for a good reason? Justify your answer.
c) In ASCII, what value represents the letter w?
d) Is the value assigned to the letter w arbitrary, or was it selected for a good reason? Justify your answer.
e) In ASCII, what value represents the character ‘7’?
f) Is the value assigned to the letter 7 arbitrary? Justify your answer.
g) In ASCII, what value represents the character ‘+’?
h) Is the value assigned to the ‘+’ character arbitrary? Justify your answer.
i) What is UTF-8?

j) What advantages does UTF-8 have compared to ASCII?
k) What disadvantages does UTF-8 have compared to ASCII?
l) Why are floating point numbers only an approximation of real numbers?

Part 3: Parsons Problem Practice

On this year's midterm exam your programming ability will be tested using a code rearrangement problem known as a Parsons Problem, and your solution to the problem will be recorded on a bubble sheet. This part of the assignment gives you the opportunity to practice both such a problem and encoding your answer on a bubble sheet. Additional practice problems can be found on the course website.

Using the lines below, create a program that reads an integer from the user and reports whether the integer is negative, zero, or positive. Some lines may not be required for a correct solution. Lines may be used multiple times.

```
1: if n < 0:
2: if n <= 0:
3: if n = 0:
4: if n == 0:
5: if n > 0:
6: if n >= 0:
7: message = "negative"
8: message = "positive"
9: message = "zero"
12: message = negative
13: message = positive
14: message = zero
15: n = input(int("Enter an integer: "))
16: n = int(input("Enter an integer: "))
17: print("The entered number was %s." % message)
18: print(message)
19: print(negative)
23: print(positive)
24: print(zero)
```

Encode your answer to the problem on the bubble sheet by writing your name and student number, shading the bubbles for your student number, writing the Python statements for your solution, writing each line number, and shading the bubbles for each line number. When a line number consists of only a single digit shade only that bubble. When the line number consists of two digits shade the bubbles for both digits. Bubble sheets can be picked up in class or printed from the course website (using a high quality printer).

Grading:

Part one of the assignment will be graded out of 50, with the grade based on the program's level of functionality and conformance to the specifications. A small number of bonus marks may be awarded to particularly impressive submissions. Part two of the assignment will be graded based on the number of correct answers provided. It is out of 44. Part 3 of the assignment is out of 6 and will be graded based on how close your answer is to a correct solution to the problem.

The total mark achieved for the assignment will be translated to a letter grade using the following table:

Mark	**Letter Grade**
100+	A+
95 to 99	A
90 to 94	A-
87 to 89	B+
83 to 86	B
80 to 82	B-
77 to 79	C+
73 to 76	C
70 to 72	C-
65 to 69	D+
60 to 64	D
0 to 59	F

As a reminder, the University of Calgary assigns the following meaning to letter grades:

A: Excellent – Superior performance showing a comprehensive understanding of the subject matter

B: Good – Clearly above average performance with generally complete knowledge of the subject matter

C: Satisfactory – Basic understanding of the subject matter

D: Minimal Pass – Marginal performance; Generally insufficient preparation for subsequent courses in the same subject

F: Fail – Unsatisfactory performance