Simon Fraser University
Computing Science 120
PRACTICE Midterm Examination

Time: X minutes

Last/Family Name (please, print): ____________________________

First Name (please, print): ____________________________

Student Number: ____________________________

Signature: ____________________________

Instructor: Anne Lavergne

This examination has X pages inclusive.
Verify that you have a complete paper.

Instructions:

✓ One cheat sheet allowed. This cheat sheet must …
  • be 8 ½ by 11 inches and one-sided,
  • be handwritten,
  • have your name and student number on its top right corner, and
  • be handed in with this midterm examination.

✓ No books, calculators, computers, cell phones, or other materials may be used.

✓ Read each question carefully before answering it.

✓ You must use Python 3. Always comment your code and always provide a useful user interaction.

✓ You can assume that all Python code fragments given in this exam are syntactically correct, unless stated.

✓ The marks for each question are given in [ ].
Use this to manage your time.

Good luck!

Q # | Marks
--- | ---
1 | / 2
2 | / 9
3 | / 4
4 | / 10
Total | / 25
Please, answer on this exam paper.

1. [2 marks] Write the header of a Python function which, given 3 numbers, figures out the smallest number. The 3 numbers are not entered by the user within the function.
   **Warning:** Marks will be deducted if you write more than what the question is asking for.

   ```python
   def smallestOfThree( firstNum, secondNum, thirdNum ) :
   ```
2. a) [6 marks] **Problem Statement:** Write a **Python function** that takes a string of float values as a parameter and returns a float value. The float values contained in the string represent prices and the returned float value represents the total of these prices.

For example, if the string is "2.35 0.75 18.10 3.75", then the function returns **24.95** and if the string is "1.75 34.97 33.10", the function returns **69.82**. Note that the prices are separated by one white space character.

You can assume that the strings passed to your function will always be composed of prices as exemplified above or be empty.

If you find it useful to start by designing an algorithm, feel free to do so.

# This function sums the prices (float number) contained in the string prices and returns it.
def sumPrices( prices ):
    # Extract each price from the string into a list
    listOfPrices = prices.split()
    aSum = 0.00
    # Add each price to a running sum
    for aPrice in listOfPrices:
        # Convert each price (a string) into a float
        # then add it to the running sum
        aSum += float(aPrice)
    return aSum

------------
below is not part of the answer-------------

# Main
thePrices = "2.35 0.75 18.10 3.75"
print("Expected result aSum = 24.95 and aSum is indeed = %0.2f\n" %sumPrices( thePrices ))
thePrices = "6.52"
print("Expected result aSum = 6.52 and aSum is indeed = %0.2f\n" %sumPrices( thePrices ))
thePrices = "1.75 34.97 33.10"
print("Expected result aSum = 69.82 and aSum is indeed = %0.2f\n" %sumPrices( thePrices ))
thePrices = ""
print("Expected result aSum = 0.00 and aSum is indeed = %0.2f\n" %sumPrices( thePrices ))
b) [3 marks] List the test data you would use to completely test the function you created in 2. a) (on the previous page) and list the corresponding results you would be expecting.

<table>
<thead>
<tr>
<th>Test Data</th>
<th>Expected Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;2.35 0.75 18.10 3.75&quot;</td>
<td>24.95</td>
</tr>
<tr>
<td>&quot;&quot;</td>
<td>0.00</td>
</tr>
<tr>
<td>&quot;6.52&quot;</td>
<td>6.52</td>
</tr>
</tbody>
</table>
3. [4 marks] Consider the following Python code fragment:

```python
def spell(theWord):
    theLength = len(theWord)
    index = 1
    for letter in theWord:
        print("Letter %i is: %s" %(index, letter))
        index = index + 1
    return theLength

# Main
aWord = input("Please, enter a word: ")
print("The word you entered is %s" %aWord)
aLength = spell(aWord)
print("This word has %i letters." %aLength)
```

Write what this Python code fragment produces when it is executed and the user enters 264. If the Python code fragment produces an error, write “Error” and explain what causes the error.

**Please, enter a word: 264**

**The word you entered is 264**

**Letter 1 is: 2**

**Letter 2 is: 6**

**Letter 3 is: 4**

**This word has 3 letters.**

Extra:

If you mentioned that the code suffers from semantic error since

1. “264” cannot be consider a word, as indicated by the variable “aWord” or the parameter “theWord”, OR

2. The elements of the string “264” cannot be considered letters, as indicated by the iterating variable “letter” and the words “Letter” and “letters” in the output printed by the program. These elements are characters.

-> Good for you! 😊

However, this semantic error does not stop the code from execute.
4. [10 marks] **Problem Statement:** Write a **complete Python program** that contains one function called `replaceEven(...)`, which requires a number `num` and a character `symbol`. With these, the function builds a string by replacing the even integers (from 1 to `num`, inclusively) with the character `symbol`.

For example, if `num = 9` and `symbol = “^”`, the `replaceEven(...)` function creates and prints the string "1^3^5^7^9", and if `num = 12` and `symbol = “!”`, the `replaceEven(...)` function creates and prints the string "1!3!5!7!9!11!".

Once the function `replaceEven(...)` has created and printed a string, it returns the number of time it replaced an even number with the character `symbol`. The Main section of your program then prints this number.

If you find it useful to start by designing an algorithm, feel free to do so.

```python
# Header comment block:
# <name of file>.py
# Description of program
# Author
# Date

# This function replaces even numbers of the sequence 1 to # num with symbol. This function also counts how many times # it replaces and returns this count.
def replaceEven(num, symbol):
    aString = ""     # initialize aString
count = 0         # initialize count
    # builds a string using the sequence 1 to num (inclusively)
    for number in range(1, num+1):
        # is number an even number?
        if number % 2 == 0:
            aString += symbol     # if so, replace it with symbol
            count += 1            # once more replacement
        else:
            # if not, number goes into the string
            aString += str(number)
    # print the built string
    print("Function replaceEven(%d,%s) produced this string
's'.\n"%(num, symbol, aString))
```
# return the number of replacements made
    return count

# Main part of the program
# Test case #1 - set num and symbol
    num = 9
    symbol = '^'
# Test case #1 - call replaceEven() passing num and symbol
# as parameters and print the number of replacements
# Label the output well (helpful user interaction)
    print( "Function replaceEven(%d,%s) replaced %d even numbers by
the character '%c'.\n" % (num, symbol, replaceEven(num, symbol), symbol))

# Test case #2 - set num and symbol
    num = 12
    symbol = '!
# Test case #2 - call replaceEven() passing num and symbol
# as parameters and print the number of replacements
# Label the output well (helpful user interaction)
    print( "Function replaceEven(%d,%s) replaced %d even numbers by
the character '%c'.\n" % (num, symbol, replaceEven(num, symbol), symbol))