Cairo University, Faculty of Computers
and Artificial Intelligence

FACULTY OF COMPUTERS AND ARTIFICAL INTELLIGENCE, CAIRO UNIVERSITY

CS111: Fundamentals of CS
Fall 2021

Sheet 2 – Version 7.0

Course Professors:
Dr. Mohammad El-Ramly

Revision History

Version 1.0	By Dr. Nouh Sabry	22 Oct. 2016
Version 2.0	By Dr. Mohammad El-Ramly	30 Oct. 2016
Version 3.0	By Dr. Mohammad El-Ramly	25 Oct. 2017
Version 4.0	By Dr. Eman Hossny	28 Oct. 2017
Version 5.0	By Dr. Mohammad El-Ramly	28 Oct. 2017
Version 6.0	By Dr Mohammed El-Ramly	10 Nov. 2021
Version 7.0	By Dr Mohammed El-Ramly	27 Oct. 2021

CS111: Fundamentals of CS
Sheet 2 – Version 7.0

Cairo University, Faculty of Computers
and Artificial Intelligence

Do not tell me I did not take this in lecture. Find the answers yourself.

Objectives

By solving the entire sheet completely, you will achieve the following goals:

1- Train on problem solving and developing solutions in the form of pseudo-codes or flowcharts.
2- Learn representations of different types of data inside the computer.
3- Learn Python basics and using it for problem solving
4- Study for the midterm exam.

Part I – Flowcharts, Algorithms and Python

1. Draw the flowchart design for an algorithm to find the minimum of an array **A** containing 100 integers.

2. Assume that you are given time-of-the-day in the format hours-minutes-seconds, using 24 hours format. Devise an **algorithm** to subtract one time from another. The specification is:
Input: integers h1, m1, s1 – representing a time, e.g. 13, 30, 12
h2, m2, s2 – represents another time that is prior to the first time, e.g., 11,40, 56.
Output: integers h, m, s – difference between the two times in hours-minutes-seconds format. For the examples above, this will be 1, 49. 16

3. Given a student's mark, an integer between 0 and 100, it is required to classify his grades to be (A, B, D, or F) based on the following:
- A, if the grade is in a range of 70-100
- B, if the grade is in a range of 50- 69
- D, if the grade is in a range of 40- 49
- F, if it is less than 40

Draw a flowchart diagram to calculate the degree classification D from the given mark.
Test the execution of the flowchart step-by-step on marks, 30, 41, 50, 76.
Draw the flowchart on the flowgorithm tool

4. You need to write an **algorithm** for finding the factorial of a non-negative integer n. Recall that the factorial of n is defined as the product:

$$\mathbf{1 \times 2 \times 3 \times \ldots\ldots \times (n-1) \times n}$$

The factorial of 0 is defined to be 1.
Draw a flowchart diagram for your algorithm.
Test the execution of the flowchart step-by-step on n = 0, 2, 6, 10

Implement the algorithm as a Python script.
Test the execution of the flowchart step-by-step on n = 0, 2, 6, 10

5. Write an algorithm to take a string from the user then reverse it (e.g. if it contains abbxc, then the output will be cxbba). **Implement it in Python.**

a- Update the previous algorithm to check if the given array is palindrome or not. **Implement it in Python.**

b- Trace your algorithm if the string is xyzzyx.

6. Write a while-loop to swap every successive pair of elements in an array A of 50 elements, i.e., A[0] and A[1] should be swapped, A[2] and A[3] should be swapped and so on. **Implement it as a Python Algorithm**

7. Suppose **A** is an array of 100 elements. Write a for-loop that stores the elements of A in reverse order in a new array X. That is, A[0] must go to X[99], A[1] must go to X[98] and so on.

8. Write an algorithm to delete an element *A[p]* from an array segment *A[m...n]*. The specification is:
inputs: A - an array
m, n, p - positions in the array such that $m \leq p \leq n$
effects: The element *A[p]* is deleted from the array so that the sequence *A[m…(n-1)]* contain the original values of *A[m…n]* except for *A[p]*.

9. Is there a difference between an algorithm and its representation? And what is the problem that appears if an algorithm is not represented in enough details.

10. Describe how the use of primitives helps remove ambiguities in an algorithm's representation.

11. Rewrite the following algorithm segment using a do-while loop rather than a while structure. Be sure the new version prints the same values as the original. Initialization:

```
num = 0
while (num < 50):
   if (num is Odd)
       print(num is Odd)
   num = num + 1
```

12. Rewrite the following algorithm segment using a while structure rather than a do while loop. Be sure the new version prints the same values as the original.

```
num = 100
   do:
     print(num)
     num = num - 1
   while (num > 0)
```

13. Design an algorithm for finding all the factors of a positive integer. For example, in the case of the integer 12, your algorithm should report the values 1, 2, 3, 4, 6, and 12. **Implement it in Python.**

CS111: Fundamentals of CS
Sheet 2 – Version 7.0

Cairo University, Faculty of Computers
and Artificial Intelligence

14. The following is an addition problem in traditional base 10 notation. Each letter represents a different digit. What digit does each letter represent? How did you get your foot in the door?

```
  SEND
+ MORE
 MONEY
```

15. The following program segment is designed to report which of the positive integers X and Y is a divisor of the other. Is the program segment correct? Explain your answer.

```
if (X < Y)
     if (Y % X == 0):
          print('X is Divisor of Y')
     else if (X % Y == 0):
          print('Y is Divisor of X')
```

16. Encryption is a very very important topic in computing. It means changing the data so that enemies (or bad people) do not know what it is. This is needed, for example, when transferring financial data like bank account numbers and credit card numbers through the internet.
Write an algorithm (in pseudo-code) that takes a message from the user and converts it to an encrypted version using Cesar cipher. Cesar cipher, in the simplest form, is shifting each character to the next one. So the message "abc" becomes "bcd" and the message "I love computers" become "J mpwf dpnqvufst". Hint: Think of a solution that uses the ASCII code of the letter to generate the replacement letter.

17. We want to write a program to convert a positive decimal number to the equivalent binary number in one byte. We already do this algorithm by hand, but we want to formulate it in the form of pseudo-code so we can program it in Python. Your algorithm should:

1) Take the input from the user.
2) Check if it is in the range that can be represented in one byte. If not, show an error message.
3) Convert it to binary and display the result.

Test your algorithm on these numbers: 12, -10, 127, 320. Show its execution step-by-step.

18. The German mathematician Leibniz (1646–1716) discovered the rather remarkable fact that the mathematical constant (π) *Pi* ﻠ can be computed using the following mathematical relationship:

$$\frac{\pi}{4} \cong 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \cdots$$

The formula to the right of the equal sign represents an infinite series; each fraction represents a term in that series. If you start with 1, subtract one-third, add one-fifth, and so on, for each of the odd integers, you get a number that gets closer and closer to the value of π/4 as you go along. Then we can get π.

A. Write a detailed algorithm in English or pseudo-code that solves this problem by calculating *n* terms, which the user inputs and then calculating π. **Implement it in Python.**
B. Test your program for *n* = 10, i.e., calculate 1-1/3+1/5-1/7+1/9-1/11+1/13-1/15+1/17-1/19

CS111: Fundamentals of CS
Sheet 2 – Version 7.0

Cairo University, Faculty of Computers
and Artificial Intelligence

Part II – Data Representation

19. Floating point. Decode the following bit patterns using the floating-point format descried in the book: (1 bit for sign, 3 for exponent using excess notation and 4 for mantissa represented in the normalized form explained in the book with the radix point moved to the left of the leftmost one)

a. 01011001 b. 10101100 c. 10001111

---

20. Floating point. What is the floating point number represented by the following bit patterns in the IEEE754 standard for representing floating-point numbers with single precision in 32 bits? **Explain** how you obtained your answer.

10111110**10110000**10100011**11010111** and 01000010**11111101**00000000**00000000**

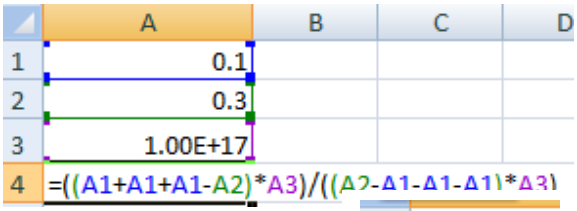
Check the correctness of your answer at:

- http://www.h-schmidt.net/FloatConverter/IEEE754.html
- http://coderstoolbox.net/number/

---

21. Truncation (Round-off) Error.

The two terms: Truncation and Round-off errors have some similarity. Although some people may use them for the same meaning, they have some differences.
Round-off error occurs when a value cannot be presented precisely inside the computer. Hence, an approximate value is used instead. It is usually used to refer to errors occurring in mathematical calculations; when the result of the calculation is not accurate because the computer could represent it precisely.
Truncation error happens when we try to fit a big value in a small variable or memory location that is not able to accommodate the whole number and the result is storing a smaller value than the original value. For example, in C language, if you try to store 5.5 in an integer variable, only 5 will be stored and 0.5 will be truncated.
In excel 2007, I tried the shown experiment on a 32 bit machine. Cell A4 has the shown formula which I expected normally to give a division by zero error #DIV/0!. But instead it gives **-2** as shown to the right.



What is the interpretation of this unexpected result? Why did this happen?

---

22. Overflow. Which of the following addition operations cause overflow? **How did** you discover? **Compare** the expected result with the actual result and explain why they are different.

A.	B.	C.	D.	E.	F.
0011	1100	1100	0100	1011	0100
+ 1010	+ 0100	+ 1100	+ 0100	+1100	+ 0101

23. ASCII.

a) Here is a message in ASCII. **What does it say**?

```
01000011 01101111 01101101 01110000
01110101 01110100 01100101 01110010
00100000 01010011 01100011 01101001
01100101 01101110 01100011 01100101
00100001
```

a) The following message is encoded in ASCII using one byte per character and then represented in hexadecimal notation. **What is it**?
436F6D7075746572 436F6D7075736572

c) Encode the following sentences in ASCII using one byte per character.
a. Is 1 byte = 8 bits?
b. Yes, a byte contains 8 bits!

---

24. Unicode.

a) **Why** was Unicode invented? **What was** the limitation of ASCII code? Compare the two from **four different** aspects.

b) Write your full name (as in the national ID) in Arabic. Then write it in hexadecimal as at it is represented inside the computer using Unicode standard for representing characters. **Explain** how you got your answer. **What** resources or web pages you used to find information. **Explain the representation in** (1) UTF-16, (2) UTF-16 Big Endian and (3) UTF-8

- https://www.branah.com/unicode-converter
- http://jrgraphix.net/r/Unicode/0600-06FF

Part III – Problem Solving and Python

25. Exercise 1: Choose all correct answer: Python is

a. is statically typed
b. is dynamically typed
c. is interpreted
d. uses type inference
e. is used only in game programming

26. Exercise 2: Assume that we execute the following assignment statements:

```
>>> width = 15
>>> height = 12.0
>>> x=-3
```

For each these expressions, write the value of the expression and the type of this value.

1. width // 2
2. x // 2

3. width /2.0
4. height / 3
5. 1 + 2 * 5

Use the Python interpreter to check your answers

27. Exercise 3: Write a Python program to calculate the sum of the digits in an integer.

28. Exercise 4: update exercise 3 to check whether a given number is an Armstrong number or not.

A positive integer is called an Armstrong number if the sum of the cubes of individual digits of the number is equal to that number itself. For example, the sum of cubes of individual digits of the number 153 is 1* 1 * 1 + 5 * 5 * 5 + 3 * 3 * 3 = 153. Hence, number 153 is an Armstrong number.

29. Exercise 5: Write a python program that takes two integer values (start and end). This program will find all numbers (between start and end) which are divisible by 9 but are not divisible by 4.

The numbers obtained should be printed in a comma-separated sequence on a single line

30. Exercise 6: Write a ***Python* program** to calculate the power of two numbers. Don't use ** operator.

31. Exercise 7: Write a ***Python* program** to take from the user three integer numbers (x,y, z) and your program should search about an operations (e.g., +, -,*,%, etc) that can be executed on both x and y such that the result should be z, then you program would print x operations y = z

Input:	Input:
2 3 8	10 5 15
Output: 2**3 = 8	Output: 10+5 =15

32. Exercise 8: Given a word of five characters in lower case, write a flowchart and pseudo code to convert this word to upper case. Example: “apple” should be “APPLE.

Hint: To get the ascii code of character 'a' for example, use ascii('a') and to convert ascii code 97 for example back to character, use character(97)

33. Exercise 9: Choose the most correct answer: Your coworker has a list called **`data`**, and is trying to pull all the data points greater than 50.0 into a new list called **`big_nums`**. He wrote the following Python program to solve this problem.

```
1  data = [1.0, 100.0, 100.0, 1.0, 100.0]
2
3  for item in data:
4      if item > 50.0:
5          big_nums.append(item)
6
7  print big_nums
```

When he runs the code above, he gets a NameError saying big_nums is not defined on line 5. He expected to see [100.0, 100.0, 100.0] printed out. How should that be fixed?

1) By replacing `big_nums.append(item)` with `big_nums = [item]` on line 5
2) By adding `big_nums = []` between line 4 and 5
3) By adding `big_nums = []` on line 2

34. **Exercise 10:** Give an example each one of these errors: (1) syntax error, (2) a run-time error, and (3) logical error.

35. **Exercise 11:** Write a ***Python program*** that will take a non-empty string and an `int n`, return a new string where the `char` at index `n` has been removed. The value of `n` will be a valid index of a char in the original string (i.e. `n` will be in the range `0..len(str)-1` inclusive)

Hint: **Do not use** loops, use string indexing, range, and concatenation

36. **Exercise 12:** Write a ***Python program*** that will take a string and check if it is palindrome or not.

37. **Exercise 13:** Write a python program to delete an element *A[p]* from an array segment *A[m...n]*. The specification is:
    Inputs: A - an array
    m, n, p - positions in the array such that $m \leq p \leq n$
    Effects: The element *A[p]* is deleted from the array so that the sequence *A[m…(n-1)]* contain the original values of *A[m…n]* except for *A[p]*.
38. **Exercise 14** Write a Python function that accepts a string and calculate the number of digits and letters.
    Sample Data: `MyPython 3.117`
    Expected Output: `Letters 8`
    `Digits 4`

---

39. **Exercise 15:** Write an algorithm and draw a flowchart to print the SUM of numbers from LOW to HIGH. Test with LOW=3 and HIGH=9.**(2)** Write a Python program to implement this algorithm.

40. **Exercise 16:** Write a python function that takes an integer number (size) then it generates Fibonacci sequence with the given size. (*Hint: The Fibonacci sequence is a sequence of numbers where the next number in the sequence is the sum of the previous two numbers in the sequence. The sequence looks like this: 1, 1, 2, 3, 5, 8, 13, …*)

    For example: if we input 3→ the output will be 1, 1, 2

    If we input 6 → the output will be *1, 1, 2, 3, 5, 8*