GEBZE
TECHNICAL UNIVERSITY

INF211

LABORATORY LEAFLET

FOR STUDENTS

LABORATORY-1
Strings, Iterations and Branching

Tasks	Explanations
1-a	Summation
1-b	Slicing
2-a	String operations
2-b	Factorial operation
3-a	Divisibility
3-b	Divisibility
4	Loops and Branching

Task 1-a: Sum up all the digits on your id number and print the result.

- *inputs: N/A*
- *outputs: integer*
-

```
my_id = "1234"
>>>
10
```

Task 1-b: Ask two numbers from the user to define indexes, then find the substring of your my_name variable taking these two index values as beginning and end. Print this substring.

- *Both of these numbers should be taken as modulo of the length of your name.*
- *Order of the numbers does not matter, slice it using minimum - maximum.*
- *Both indexes are inclusive.*
- *inputs : input1, input2 $\in \mathbb{Z}$.*
- *output : string*

```
my_name = "Tarkan"
>>>
Enter first number: 2
Enter second number: 5
Rkan
```

Task 2-a: Ask for a string from the user, then count the number of vowels in the input and print the result.

- *If the input does not have any vowels, print 0.*
- *vowels ∈{ 'a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U'}*
- *inputs: input = {x: printable characters except whitespaces and len(x) ∈*
- *[1, 100]}. (This includes letters in English, numbers and punctuation characters.)*
- *outputs: integer*

```
>>>
Enter input: inf211
1

>>>
Enter input: F!ucvh1!UjK\;'an!+!@#
3
```

Task 2-b: Ask for a number from the user, then calculate and print the factorial of that number.

- *Do NOT use any libraries or predefined functions*
- *inputs: input = {x: x ∈ ℕ and x ≤ 30}*
- *outputs: integer*

```
>>>

Enter input: 4

24
```

Task 3-a: Ask for a number from the user. If that number is divisible to both 3 and 7 print True, else print False.

- *inputs: input ∈ ℕ*
- *outputs: boolean*

```
>>>

Enter a number: 12

False

>>>

Enter a number: 21

True
```

Task 3-b: Ask for a number from the user, then find if that number is a prime number. If the number is a prime number, print True, else print False. (You can check reference [1] for this)

- *inputs: input = { x: x ∈ ℕ and x > 1}*
- *outputs: boolean*

```
>>>

Enter a number: 12

False

>>>

Enter a number: 23

True
```

Task 4: Find the square root of a given number using Heron's method (Babylonian method) that we discussed in the first lecture. (You are advised to see reference [2])

- *Do NOT use any libraries or predefined functions (i.e. sqrt)*
- *Choose an appropriate iteration number.*
- *Choose an appropriate starting guess. (i.e 1)*
- *inputs: input = { x: x ∈R and 1e9 ≥ x ≥ 0.0}*
- *outputs: float*

```
>>>

Enter a number: 16

4.0

>>>

Enter a number: 12

3.4641016151377544
```

[1] https://en.wikipedia.org/wiki/Prime_number
[2]https://en.wikipedia.org/wiki/Methods_of_computing_square_roots#Babylonian_method