Laboratory Assignment 2

Problem 1 - Fahrenheit - Celsius conversion [6 pts]

Ask for a Fahrenheit degree from the user, then convert to Celsius and return the result as a float.

- `inputs : input = { x: x ∈ R and 200.0 ≥ x ≥ -100.0}`
- `output : float (with epsilon = 1E-9)`

```
>>>
Enter Fahrenheit degree: 68
20.0
```

Problem 2 - Celsius - Fahrenheit conversion [6 pts]

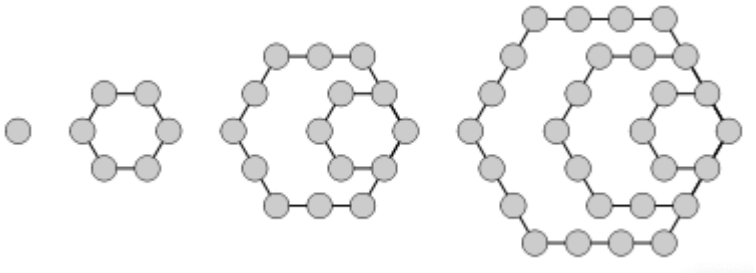
Ask for a Celsius degree from the user, then convert to Fahrenheit and return the result as a float.

- `inputs : input = { x: x ∈ R and 100.0 ≥ x ≥ -100.0}`
- `output : float (with epsilon = 1E-9)`

```
>>>
Enter Celsius degree: 20
68.0
```

Problem 3 - Hexagonal number [7 pts]

The nth hexagonal number h_n is the number of distinct dots in a pattern of dots consisting of the outlines of regular hexagons with sides up to n dots, when the hexagons are overlaid so that they share one vertex. [1]



The formula for the *n*th hexagonal number is given as:

$$h_n = 2n^2 - n$$

Ask for a number from the user. Calculate and return the hexagonal number that corresponds to that number. As an example, first 7 hexagonal numbers are: 1, 6, 15, 28, 45, 66, 91

- `inputs : input = {x: x ∈ ℕ and 1E6 ≥ x ≥ 1}`
- `output : integer`

```
>>>
Enter a number: 1
1

>>>
Enter a number: 6
66
```

Problem 4 - Lucas number [14 pts]

Each Lucas number is defined to be the sum of its two immediate previous terms and the first two Lucas numbers are L(0) = 2, and L(1) = 1 as shown below. [2]

$$L_n := \begin{cases} 2 & \text{if } n = 0; \\ 1 & \text{if } n = 1; \\ L_{n-1} + L_{n-2} & \text{if } n > 1. \end{cases}$$

Ask for a number from the user. Calculate and return the lucas number that corresponds to that number starting with index 0. As an example, first 7 Lucas numbers are: 2, 1, 3, 4, 7, 11, 18

- inputs : input = {x: x ∈ ℕ and 1E6 ≥ x ≥ 0}
- output : integer

```
>>>
Enter a number: 0
2

>>>
Enter a number: 6
18
```

Problem 5 - Reversing a string [3 pts]

Ask for an input string from the user and return the reverse of it.

- inputs : input ∈ {x: printable characters except whitespaces and len(x) ∈ [1, 100]}.
- output : string

```
>>>
Enter a string: Inf211
112fnI
```

Problem 6 - Removing unwanted printable characters [11 pts]

Ask for an input string from the user and return the string with **only letters and numbers** in the **entered order.**

- Unwanted printable characters that should be removed are: !"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~

- Note that the string can include " or ' characters, so take necessary precautions.
- `inputs : input = {x: x ∈ printable characters except whitespaces and len(x) ∈ [1, 100]}.`
- `output : string`

```
>>>
Enter a string: :I!n#f21@1;,
Inf211

>>>
Enter a string: :I!n'"#f"2%$/1@1;,.
Inf211
```

Problem 7 - Base 4 representation [13 pts]

Base 4 is a number system that represents a given number using the {0, 1, 2, 3} set.

Ask for an integer from the user, then return the base 4 representation of that number as a string.

- There should be no leading zeros for any credit: i.e: it should return "123" instead of "00123".
- If the number is negative, prepend a minus sign (-) to the string.
- `inputs: input = {x: x ∈ ℤ and 1E6 ≥ x ≥ -1E6}`
- `output: string`

```
>>>
Enter input: 14
32

Explanation: 3 * 4**1 + 2 * 4**0 = 15

>>>
Enter input: -14
-32

>>>
Enter input: 27
123

Explanation: 1 * 4**2 + 2 * 4**1 + 3 * 4**0 = 27

>>>
Enter input: -27
-123
```

Problem 8 - Valid brackets [15 pts]

Ask for an input from the user containing only parentheses. Then return if the parentheses are actually in correct order (or it is valid).

- The parentheses are valid if open brackets closed by the same type of brackets and open brackets closed in the correct order.
- inputs: input = {x: x ∈ "{,},(,),[,]" and len(x) ∈ [1, 100]}
- output: boolean

```
>>>
Enter input: {}
True

>>>
Enter input: {}{}[]
True

>>>
Enter input: [{}]
True

>>>
Enter input: {([])}[]
True

>>>
Enter input: {(})
False

Explanation: After {( expression, first ( should be closed with ), then }.

>>>
Enter input: [)
False

Explanation: After [, it should be closed with ].

>>>
Enter input: [()
False

Explanation: First [ is never closed.
```

Problem 9 - Last word length [11 pts]

Ask for a string input from the user, then find the length of the last word in the string. Return the length.

- inputs: input = {x: x ∈ All English letters and space, and len(x) ∈ [1, 200]}.
- output: integer

```
>>>
Enter a string: There are no secrets to success It is the result of preparation hard work and learning from failure
7

Explanation: Last word is "failure" and its length is 7.

>>>
Enter a string: The way to get started is to quit talking and begin doing
5

Explanation: Last word is "doing" and its length is 5.

>>>
Enter a string: Geronimo
8

Explanation: Last word is "Geronimo" and its length is 8.
```

Problem 10 - Escape the maze [14 pts]

Assume that you are in a maze, and you are given a set of commands describing how to exit from the maze using east, west, north, and south commands each denoted with first letters (e, w, n, s). Assuming you start at the origin of the x-y coordinate system (0, 0), calculate the length of the straight line connecting your starting point and where you arrived in the coordinate plane.[3]

- Do not use any extra libraries.
- Use any square root method you choose and implement (heron, bisection, newton-raphson), but make sure you give accurate results (diff < epsilon) with your method.
- inputs: input = { x: x ∈ {w,e,n,s} and len(x) ∈ [1, 200]}.
- output: float (with epsilon = 1E-9)

```
>>>
Enter the exit route: s
1.0
```

```
Explanation: s makes it (0, -1). So we ended at (0, -1) coordinates. If we
calculate the distance (hypotenuse) from origin (0, 0):  sqrt( (abs(0 - 0)**2 +
abs(-1 - 0)**2) which is sqrt(0 + 1) = sqrt(1) = 1.0

>>>
Enter the exit route: wwenn
2.23606797749979

Explanation: first w makes it (-1, 0), second w makes it (-2, 0), e makes it (-
1, 0), n makes it (-1, 1), n makes it (-1, 2). So we ended at (-1, 2)
coordinates. If we calculate the distance (hypotenuse) from origin (0, 0):
sqrt( (abs(-1 - 0)**2 + abs(2 - 0)**2) which is sqrt(1 + 4) = sqrt(5) =
2.23606797749979

>>>
Enter the exit route: sewnsnwwwnsseeewnsswnwnwwnessnswwnewsn
6.0
```

[1] https://en.wikipedia.org/wiki/Hexagonal_number
[2] https://en.wikipedia.org/wiki/Lucas_number
[3] https://en.wikipedia.org/wiki/Hypotenuse