**Machine Learning**

# ASSIGNMENT -1

**Name: Pranshu Aggarwal**
**Branch: CSE-3**
**Roll no.: 49**
**Enrollment No.: 40576802717**

**Problem Statement 1:**

Create a program that asks the user to enter their name and their age. Print out a message addressed to them that tells them the year that they will turn 100 years old.

Extras:

1. Add on to the previous program by asking the user for another number and printing out that many copies of the previous message.
2. Print out that many copies of the previous message on separate lines. (Hint: the string "\n is the same as pressing the ENTER button)

**Pseudocode:**

1. Read name, age, and n input from user

2. if age > 0, then

3. if age > 100, then

4. print "Congratulations {name}! You have already crossed 100."

5. otherwise,

6. for i:=0 to n, step by 1,

7. print "Hey {name}, {100-age} years left to reach 100..\n"

8. end for loop

9. otherwise, print "Invalid input!"

**Program Screenshot:**

```
name = input("Enter your name: ")
age = int(input("Enter your age: "))
num = int(input("Enter a number: "))
if age > 0:
    if age > 100: print("Congratulations {name}! You have already crossed 100.")
    else:
        for i in range(num): print(f"Hey {name}, {100-age} years left to reach 100..\n", end='')
else: print("Invalid input!")
```

**Output:**

```
Enter your name: Pranshu
Enter your age: 21
Enter a number: 5
Hey Pranshu, 79 years left to reach 100..
Hey Pranshu, 79 years left to reach 100..
Hey Pranshu, 79 years left to reach 100..
Hey Pranshu, 79 years left to reach 100..
Hey Pranshu, 79 years left to reach 100..
```

**GitHub Link:** Click here for ipynb notebook

**Problem Statement 2:**

Take a list, say for example this one:

a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]

and write a program that prints out all the elements of the list that are less than 5.

Extras:

1. Instead of printing the elements one by one, make a new list that has all the elements less than 5 from this list in it and print out this new list.
2. Write this in one line of Python.
3. Ask the user for a number and return a list that contains only elements from the original list a that are smaller than that number given by the user.

**Pseudocode:**

1. Read a list 'lst'
2. for each item in lst, do
3. if item < 5, then
4. print item
5. end for loop

**Pseudocode for Extras 1&2:**

1. print list, created using list comprehension as [item for item in lst if item < 5])

**Pseudocode for Extras 3:**

1. Read n
2. Print list, created using list comprehension as [item for item in lst if item < n]

**Program Screenshot and Output:**

Main Program:

```
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
for i in a:
    if i < 5: print(f"{i} ", end='')
```

```
1 1 2 3
```

Extras 1 and 2 Program:

```
print([i for i in a if i < 5])
```

```
[1, 1, 2, 3]
```

Extras 3 Program:

```
n = int(input("Enter a number : "))
print([i for i in a if i < n])
```

```
Enter a number : 30
[1, 1, 2, 3, 5, 8, 13, 21]
```

**GitHub Link:** Click here for ipynb notebook

**Problem Statement 3:**

W.A.P. that asks the user "how many Fibonacci numbers to generate" and then generates them. Take this opportunity to think about how you can use functions. Make sure to ask the user to enter the number of numbers in the sequence to generate. (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, …)

**Pseudocode of fib(n):**

1. Declare and Initialize a list 'fibs' with 0 at $0^{th}$ index and 1 at $1^{st}$ index.
2. for i:=2 to n, step by 1, do
3. Append Sum of fibs[i-1] and fibs[i-2]
4. end for loop
5. return fibs

**Program Screenshot and Output:**

```
def fib(n):
    fibs = [0, 1]
    for i in range(2, n): fibs.append(fibs[i - 1] + fibs[i - 2])
    return fibs
n = int(input("How many fibonacci numbers to generate? Enter a number: "))
print(fib(n))
```

```
How many fibonacci numbers to generate? Enter a number: 10
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

**GitHub Link:** Click here for ipynb notebook

**Problem Statement 4:**

Write a program (function!) that takes a list and returns a new list that contains all the elements of the first list minus all the duplicates.

Extras:

- Write two different functions to do this - one using a loop and constructing a list, and another using sets.

**Pseudocode using loop:**

1. Declare and Initialize an empty list 'new_list'

2. for each item in lst, do

3. if item not in new_list, then

4. Append item to new_list

5. end for loop

6. return new_list

**Pseudocode using set:**

1. Create a set variable and store set(lst)

2. return list(set)

**Program screenshot and Output:**

```
# Using loop with change in order
def remove_duplicates_using_loop(lst):
    new_list = []
    for i in lst:
        if i not in new_list: new_list.append(i)
    return new_list

a = [1, 1, 2, 3, 5, 8, 8, 13, 21, 34, 34, 55, 89]
print(remove_duplicates_using_loop(a))
```

```
[1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

Extras Program:

```
# Using set but order changes
def remove_duplicates_using_set(lst):
    return list(set(lst))

a = [1, 1, 2, 3, 5, 8, 8, 13, 21, 34, 34, 55, 89]
print(remove_duplicates_using_set(a))
```

```
[1, 2, 3, 34, 5, 8, 13, 21, 55, 89]
```

**GitHub Link:** Click here for ipynb notebook

**Problem Statement 5:**

Ask the user for a number and determine whether the number is prime or not. (For those who have forgotten, a prime number is a number that has no divisors.). Use functions.

**Pseudocode(n):**

1. if n == 2, then, return True

2. if n is even, then, return False

3. for i:=3 to sqrt(n) + 1, step by 1, do

4. if n % i == 0, then, return False

5. end for loop

6. return True

**Program screenshot and Output:**

```
def isPrime(n):
    if n == 2: return True
    if n % 2 == 0: return False
    for i in range(3, int(n**0.5) + 1, 2):
        if n % i == 0: return False
    return True

n = int(input("Enter a number: "))
print(f"{n} is {'prime' if isPrime(n) else 'not prime'}")
```

```
Enter a number: 15
15 is not prime
```

**GitHub Link:** Click here for ipynb notebook