NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTION, AFFILIATED TO VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM, APPROVED BY AICTE & GOVT.OF KARNATAKA

Ability Enhancement Course - Python Programming

Submitted in partial fulfilment of the requirement for the award of Degree of

Bachelor of Engineering

in

Artificial Intelligence and Machine Learning

Submitted by:

Anvith skandha hedge **1NT21AI007**

Submitted to:

Mr V Sunil Kumar

1. Write a Program to print the following pattern.

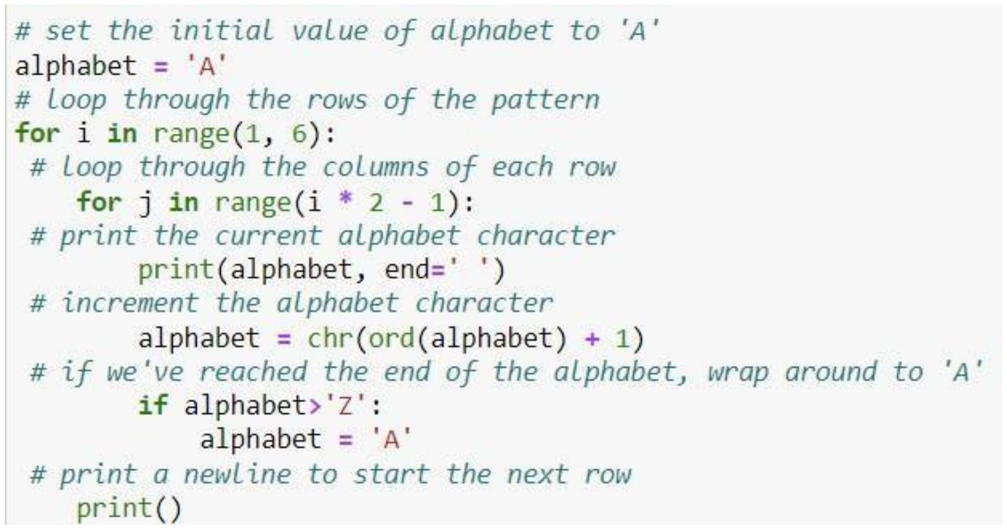
   A

   B C D

   E F G H I

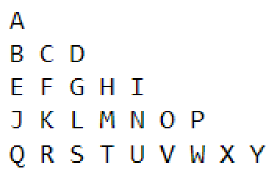
   J K L M N O P

   Q R S T U V W X Y



```
# set the initial value of alphabet to 'A'
alphabet = 'A'
# loop through the rows of the pattern
for i in range(1, 6):
 # loop through the columns of each row
    for j in range(i * 2 - 1):
 # print the current alphabet character
        print(alphabet, end=' ')
 # increment the alphabet character
        alphabet = chr(ord(alphabet) + 1)
 # if we've reached the end of the alphabet, wrap around to 'A'
        if alphabet>'Z':
            alphabet = 'A'
 # print a newline to start the next row
    print()
```

OUTPUT:



```
A
B C D
E F G H I
J K L M N O P
Q R S T U V W X Y
```

2. Given a list of strings, count and print the number of strings where the string length is 2 or more & the 1st & last characters are same.

```python
def count_strings(lst):
    count = 0
    for s in lst:
        if len(s) >= 2 and s[0] == s[-1]:
            count += 1
    return count
# example usage
strings = ['racecar', 'apple', 'civic', 'python', 'madam']
count = count_strings(strings)
print(f"There are {count} strings that meet the criteria.")
```

OUTPUT:

```
There are 3 strings that meet the criteria.
```

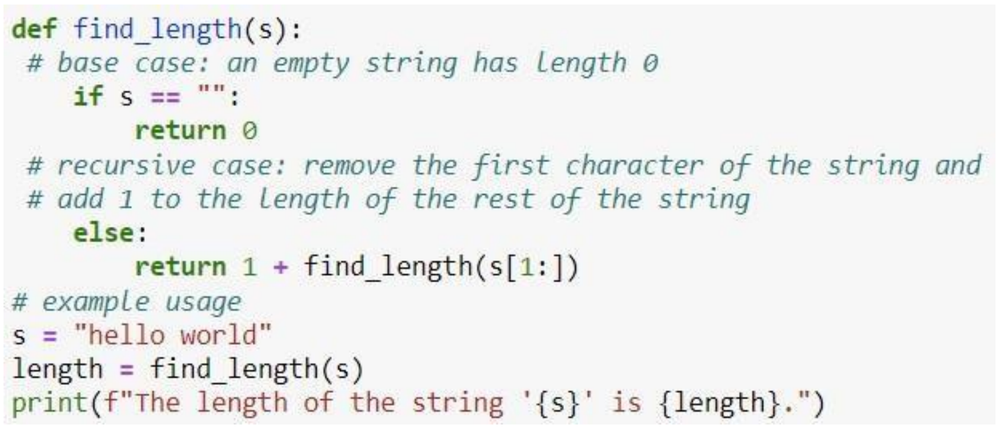
3. Write a python program to accept a sequence of whitespace separated words as input and prints the words after removing all duplicate words and sorting them alphanumerically.

```python
def remove_duplicates_and_sort(sentence):
 # split the sentence into a list of words
    words = sentence.split()
 # create a set of unique words
    unique_words = set(words)
 # convert the set back to a list and sort it
    sorted_words = sorted(list(unique_words))
 # join the sorted words back into a sentence and return it
    return ' '.join(sorted_words)
# example usage
sentence = input("Enter a sentence: ")
result = remove_duplicates_and_sort(sentence)
print(result)
```

OUTPUT:

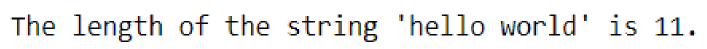
```
Enter a sentence: hello world world again zebra python snake
again hello python snake world zebra
```

4. Solve the following using Recursion:
   - find the length of a string
   - find the smallest element in a list

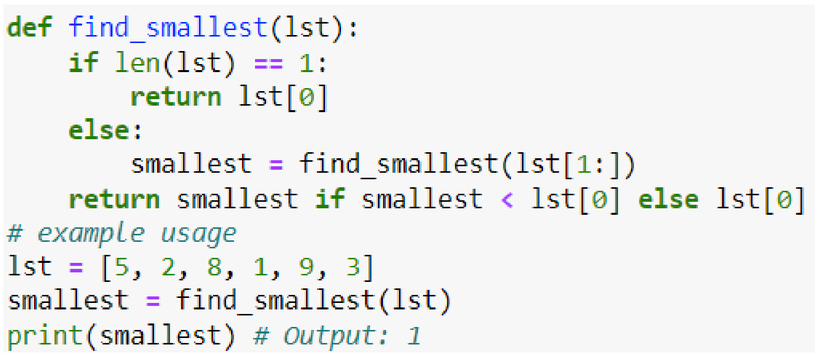


```
def find_length(s):
 # base case: an empty string has length 0
    if s == "":
        return 0
 # recursive case: remove the first character of the string and
 # add 1 to the length of the rest of the string
    else:
        return 1 + find_length(s[1:])
# example usage
s = "hello world"
length = find_length(s)
print(f"The length of the string '{s}' is {length}.")
```

OUTPUT:



```
The length of the string 'hello world' is 11.
```



```
def find_smallest(lst):
    if len(lst) == 1:
        return lst[0]
    else:
        smallest = find_smallest(lst[1:])
    return smallest if smallest < lst[0] else lst[0]
# example usage
lst = [5, 2, 8, 1, 9, 3]
smallest = find_smallest(lst)
print(smallest) # Output: 1
```

OUTPUT:

```
1
```

5. Python program to copy odd lines from one file to another file.

   Input.txt

```
hello
world
again
python snake
zebra
```

```python
def copy_odd_lines(from_file, to_file):
    with open(from_file, 'r') as f1, open(to_file, 'w') as f2:
        lines = f1.readlines()
        odd_lines = [line for i, line in enumerate(lines) if i % 2 != 0]
        f2.writelines(odd_lines)
        print("copy successful")
# example usage
from_file = 'input.txt'
to_file = 'output.txt'
copy_odd_lines(from_file, to_file)
```

OUTPUT:

```
copy successful
```

Output.txt

```
world
python snake
```

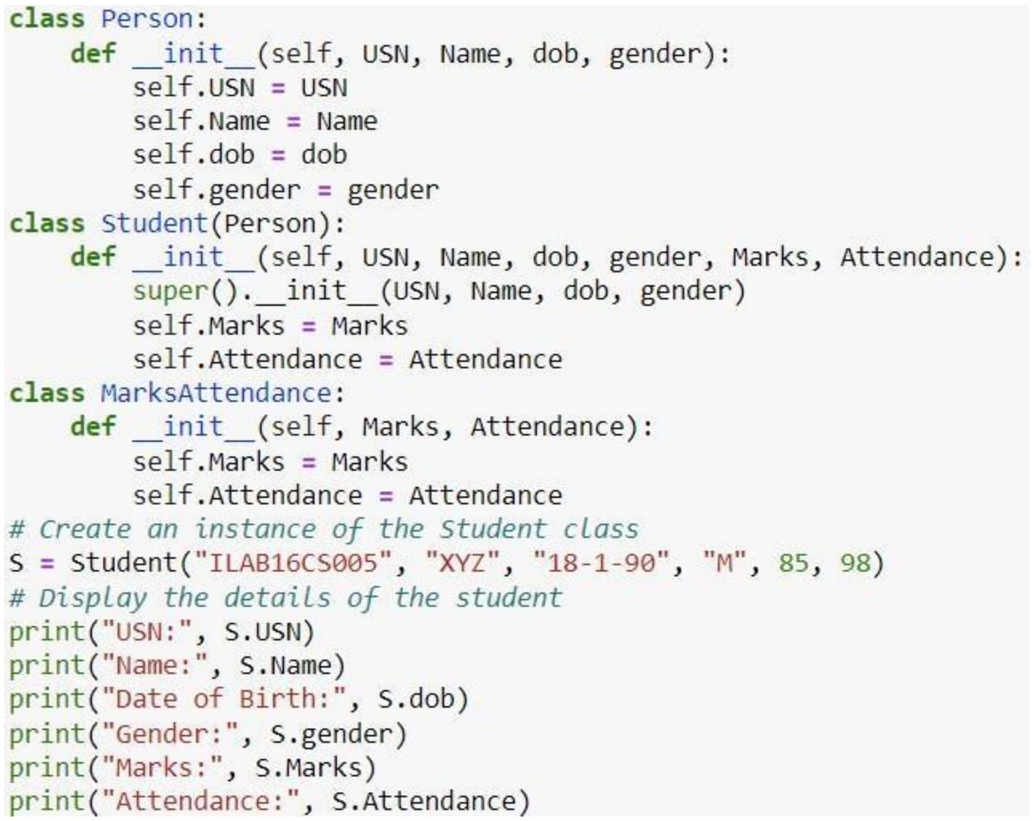
6. Python program to print the number of lines, words and characters present in the given file.

```python
def count_lines_words_characters(file_path):
    line_count = 0
    word_count = 0
    character_count = 0
    with open(file_path, 'r') as file:
        for line in file:
            line_count += 1
            words = line.split()
            word_count += len(words)
            character_count += len(line)
    print("Number of lines:", line_count)
    print("Number of words:", word_count)
    print("Number of characters:", character_count)
# Example usage:
file_path = "input.txt" # Replace with the actual file path
count_lines_words_characters(file_path)
```

OUTPUT:

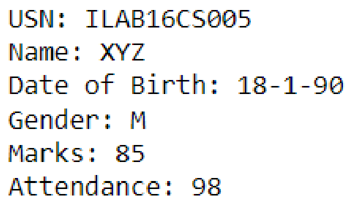
```
Number of lines: 5
Number of words: 6
Number of characters: 36
```

7. Write a program that has a class Person, inherit a class Student from Person which is has a class MarksAttendance. Assume the attributes for Person class as: USN, Name, dob, gender. Attributes for Student class as: class, branch, year, MA.Attributes for Marks Attendance: Marks, Attendance. Create a student S = Student("ILAB16CS005","XYZ","18-1-90","M",85,98) and display the details of the student.



```python
class Person:
    def __init__(self, USN, Name, dob, gender):
        self.USN = USN
        self.Name = Name
        self.dob = dob
        self.gender = gender
class Student(Person):
    def __init__(self, USN, Name, dob, gender, Marks, Attendance):
        super().__init__(USN, Name, dob, gender)
        self.Marks = Marks
        self.Attendance = Attendance
class MarksAttendance:
    def __init__(self, Marks, Attendance):
        self.Marks = Marks
        self.Attendance = Attendance
# Create an instance of the Student class
S = Student("ILAB16CS005", "XYZ", "18-1-90", "M", 85, 98)
# Display the details of the student
print("USN:", S.USN)
print("Name:", S.Name)
print("Date of Birth:", S.dob)
print("Gender:", S.gender)
print("Marks:", S.Marks)
print("Attendance:", S.Attendance)
```

OUTPUT:



```
USN: ILAB16CS005
Name: XYZ
Date of Birth: 18-1-90
Gender: M
Marks: 85
Attendance: 98
```

8. Write a python program to express instances as return values to define a class RECTANGLE with members width, height, corner_x, corner_y and member function to find centre, area, and perimeter of a rectangle.

```python
class Rectangle:
    def __init__(self, width, height, corner_x, corner_y):
        self.width = width
        self.height = height
        self.corner_x = corner_x
        self.corner_y = corner_y

    def get_center(self):
        center_x = self.corner_x + self.width / 2
        center_y = self.corner_y + self.height / 2
        return center_x, center_y

    def get_area(self):
        return self.width * self.height

    def get_perimeter(self):
        return 2 * (self.width + self.height)


# Example usage:
rect = Rectangle(5, 10, 0, 0)
center = rect.get_center()
area = rect.get_area()
perimeter = rect.get_perimeter()

print("Center:", center)
print("Area:", area)
print("Perimeter:", perimeter)
```

OUTPUT:

```
Center: (2.5, 5.0)
Area: 50
Perimeter: 30
```

9. Program to get all phone numbers of redbus.in by using web scraping and regular expressions.

```
import re,urllib
import urllib.request
u=urllib.request.urlopen("https://www.redbus.in/info/redcare")
text=u.read()
numbers=re.findall("[0-9-]{7}[0-9-]+",str(text),re.I)
for n in numbers:
    print(n)
```

OUTPUT:

```
65-31582888
2023-07-17
```

10. Write a python program to extract all mobile numbers present in input.txt where numbers are mixed with normal text data.

Input.txt

```
hello
world
again
9284628473
python snake
0011284569
zebra
```

```python
import re
f1=open("input.txt","r")
f2=open("output.txt","w")
for line in f1:
    list=re.findall("[7-9]\d{9}",line)
    for n in list:
        f2.write(n+"\n")
print("Extracted all Mobile Numbers into output.txt")
f1.close()
f2.close()
```

OUTPUT:

```
Extracted all Mobile Numbers into output.txt
```

Output.txt

```
9284628473
```

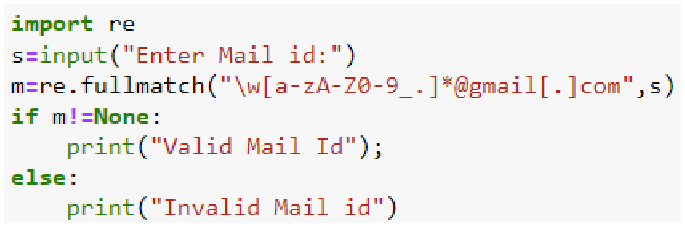
11. Write a Python Program to check whether the given number is a valid mobile number or not.

```python
import re
n=input("Enter number:")
m=re.fullmatch("[7-9]\d{9}",n)
if m!= None:
    print("Valid Mobile Number")
else:
    print("Invalid Mobile Number")
```

OUTPUT:

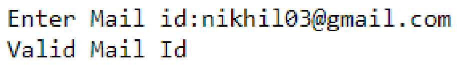
```
Enter number:11038293820
Invalid Mobile Number

Enter number:9113729375
Valid Mobile Number
```

12. Write a Python Program to check whether the given mail id is valid Gmail id or not.

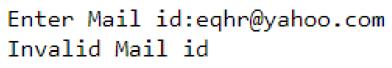


```
import re
s=input("Enter Mail id:")
m=re.fullmatch("\w[a-zA-Z0-9_.]*@gmail[.]com",s)
if m!=None:
    print("Valid Mail Id");
else:
    print("Invalid Mail id")
```

OUTPUT:

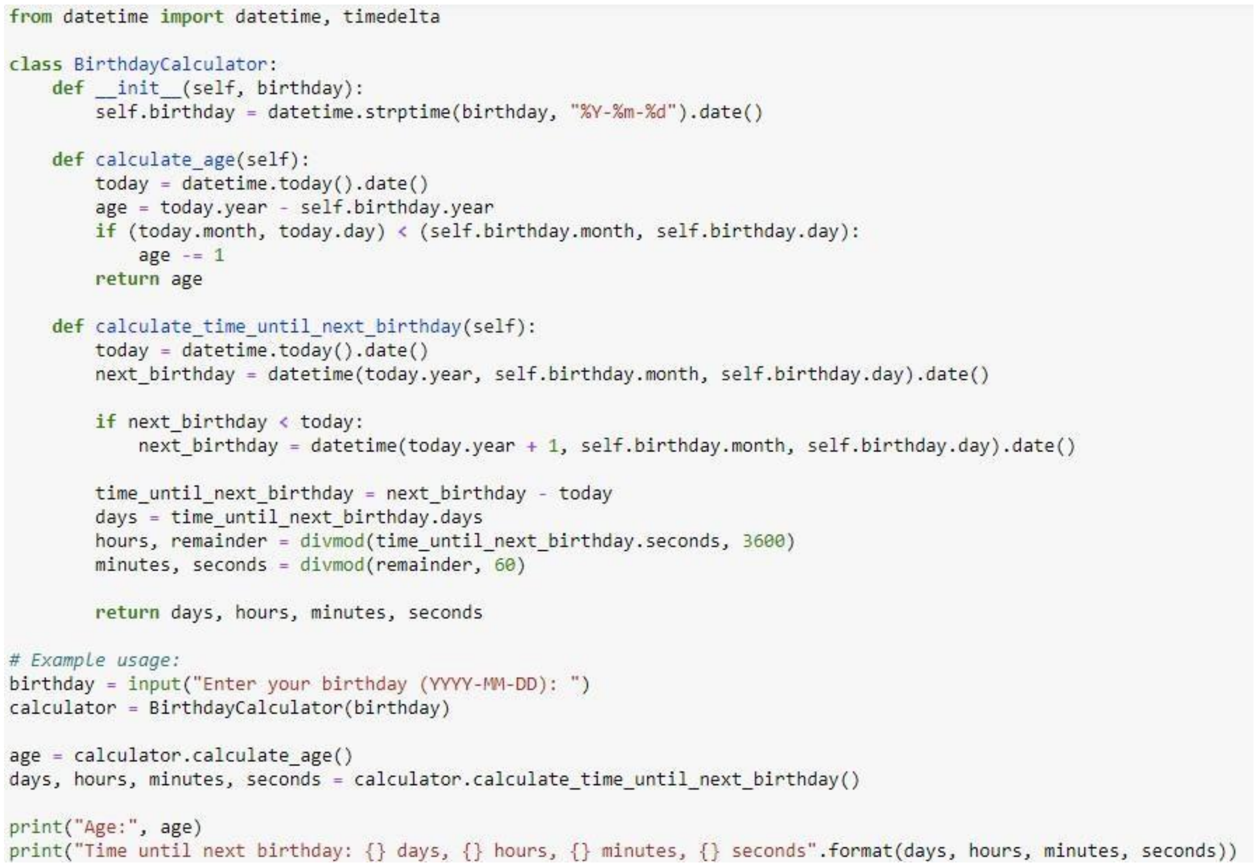


```
Enter Mail id:nikhil03@gmail.com
Valid Mail Id
```



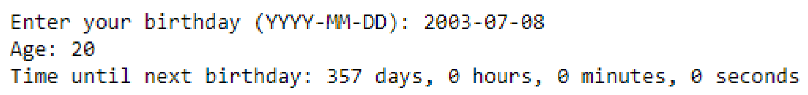
```
Enter Mail id:eqhr@yahoo.com
Invalid Mail id
```

13. Write a python program that uses date time module within a class, takes a birthday as input and prints the age and the number of days, hours, minutes and seconds until the next birthday.



```python
from datetime import datetime, timedelta

class BirthdayCalculator:
    def __init__(self, birthday):
        self.birthday = datetime.strptime(birthday, "%Y-%m-%d").date()

    def calculate_age(self):
        today = datetime.today().date()
        age = today.year - self.birthday.year
        if (today.month, today.day) < (self.birthday.month, self.birthday.day):
            age -= 1
        return age

    def calculate_time_until_next_birthday(self):
        today = datetime.today().date()
        next_birthday = datetime(today.year, self.birthday.month, self.birthday.day).date()

        if next_birthday < today:
            next_birthday = datetime(today.year + 1, self.birthday.month, self.birthday.day).date()

        time_until_next_birthday = next_birthday - today
        days = time_until_next_birthday.days
        hours, remainder = divmod(time_until_next_birthday.seconds, 3600)
        minutes, seconds = divmod(remainder, 60)

        return days, hours, minutes, seconds

# Example usage:
birthday = input("Enter your birthday (YYYY-MM-DD): ")
calculator = BirthdayCalculator(birthday)

age = calculator.calculate_age()
days, hours, minutes, seconds = calculator.calculate_time_until_next_birthday()

print("Age:", age)
print("Time until next birthday: {} days, {} hours, {} minutes, {} seconds".format(days, hours, minutes, seconds))
```

OUTPUT:



```
Enter your birthday (YYYY-MM-DD): 2003-07-08
Age: 20
Time until next birthday: 357 days, 0 hours, 0 minutes, 0 seconds
```