Computer Record

Program 1.

Aim.

Write a program to accept string as an input and to count and display the total number of times a character is present.

Program.

Program 2.

Aim.

Write a program Input a list of elements, create an empty list and add elements divisible by 5 to it. Display the resulting list?

Program.

Program 3.

Aim.

Write a program to read a number and check whether it is a prime number.

Program.

Program 4.

Aim.

Write a program to implement all the basic operations of a stack.

Program.

Program 5.

Aim.

Write a program to Create add_client and delete_client methods in Python to add a new client and remove a client from a list of client names, simulating the insertion and deletion operations of a stack data structure.

Program 6.

Aim.

Write a program to read a text file by line and display each word separated by a "#"

Program.

Program 7.

Aim.

Write a program to read a text file and display the number of vowels, consonants, upper case, lower case characters in file.

Program.

Program 8.

Aim.

Write a program to remove all the lines that contain the character “a” in a file and write it to another file.

Program.

Program 9.

Aim.

Write a program to create a binary file with name and roll number. Search for a given roll number and display the name, if not found display appropriate message.

Program.

Program 10.

Aim.

Write a program to create a binary file with roll number, name and mark. Input a roll number and update the marks.

Program.

Program 11.

Aim.

Write a program to create a random number generator that generates random numbers between 1 to 6 simulating a die.

Program.

Source Codes

Program 1:

Program 2:

Program 3:

Program 4:

Program 5:

Program 6:

Program 7:

Program 8:

Program 9:

Program 10:

Program 11:

Program 1.

Aim.

Write a program to accept string as an input and to count and display the total number of times a character is present.

Program.

```
def countchar(sentence, character):
    sentence = sentence.lower()
    character = character.lower()
    count = 0
    for char in sentence:
        if character == char:
            count += 1
        else:
            continue
    if count == 0:
        print(f"{character} is not found")
    else:
        print(f"The no of occurrence of {character} is {count}")


input_sentence = input("Enter a sentence: ")
input_character = input("Enter a character to count: ")
countchar(input_sentence, input_character)
```

Output.

```
Enter a sentence: Python is a high-level programming language designed for readability.
Enter a character to count: i
The no of occurrence of i is 6
```

Program 2.

Aim.

Write a program Input a list of elements, create an empty list and add elements divisible by 5 to it. Display the resulting list?

Program.

```
def div5():
    list_1, list_2 = [], []
    n = int(input("Enter the no of inputs: "))
    for i in range(1, n+1):
        element = int(input(f"Enter element {i}: "))
        list_1.append(element)
    print(f"List 1: {list_1}")

    for j in list_1:
        if j % 5 == 0:
            list_2.append(j)
    print(f"List of number divisible by 5: {list_2}")
```

Output.

```
Enter the no of inputs: 5
Enter element 1: 55
Enter element 2: 25
Enter element 3: 18
Enter element 4: 11
Enter element 5: 3
List 1: [55, 25, 18, 11, 3]
List of number divisible by 5: [55, 25]
```

Program 3.

Aim.

Write a program to read a number and check whether it is a prime number.

Program.

```
def prime(no):
    chk = 0
    for i in range(2, (no//2) + 1):
        if no % i == 0:
            chk += 1
    return chk


while True:
    x = int(input("Enter a number: "))
    is_prime = prime(x)

    if is_prime != 0:
        print("It is not a prime number!")
    else:
        print("It is a prime number")
```

Output.

```
Enter a number: 5
It is a prime no
Enter a number: 55
It is not a prime no
```

Program 4.

Aim.

Write a program to implement all the basic operations of a stack.

Program.

```
def push(lst):
    x = input("Enter an element: ")
    lst.append(x)
    return True


def pop(lst):
    print("Deleted element: ", lst.pop() if lst else "Stack underflow")
    return True


def peek(lst):
    print("Top element: ", lst[-1] if lst else "Stack underflow")
    return True


def display(lst):
    print("Stack elements: ", lst[::-1] if lst else "Stack underflow")
    return True


def exit(lst):
    print("Program exited Successfully")
    return False
```

```
stack = []
actions = {
    1: push,
    2: pop,
    3: display,
    4: peek,
    5: exit
}

run = True
while run:
    print('''
    1: Push
    2: Pop
    3: Display
    4: Peek
    5: Exit''')
    choice = int(input("Enter your choice (1,2,3,4,5): "))
    action = actions.get(choice)
    run = action(stack)
```

Output.

```
    1: Push
    2: Pop
    3: Display
    4: Peek
    5: Exit
Enter your choice (1,2,3,4,5): 1
Enter an element: Hi

    1: Push
    2: Pop
    3: Display
    4: Peek
    5: Exit
Enter your choice (1,2,3,4,5): 1
Enter an element: Hello

    1: Push
    2: Pop
    3: Display
    4: Peek
    5: Exit
Enter your choice (1,2,3,4,5): 1
Enter an element: 18
```

```
    1: Push
    2: Pop
    3: Display
    4: Peek
    5: Exit
Enter your choice (1,2,3,4,5): 3
Stack elements:  ['18', 'Hello', 'Hi']

    1: Push
    2: Pop
    3: Display
    4: Peek
    5: Exit
Enter your choice (1,2,3,4,5): 4
Top element:  18

    1: Push
    2: Pop
    3: Display
    4: Peek
    5: Exit
Enter your choice (1,2,3,4,5): 2
Deleted element:  18
```

```
    1: Push
    2: Pop
    3: Display
    4: Peek
    5: Exit
Enter your choice (1,2,3,4,5): 3
Stack elements:  ['Hello', 'Hi']

    1: Push
    2: Pop
    3: Display
    4: Peek
    5: Exit
Enter your choice (1,2,3,4,5): 5
Program exited Successfully
```

Program 5.

Aim.

Write a program to Create add_client and delete_client methods in Python to add a new client and remove a client from a list of client names, simulating the insertion and deletion operations of a stack data structure.

```python
def add_client(lst):
    x = input("Enter an client name: ")
    lst.append(x)
    return True


def delete_client(lst):
    print("Deleted Client: ", lst.pop() if lst else "Stack underflow")
    return True


def exit(lst):
    print("Program exited Successfully")
    return False
```

```python
stack = []
actions = {
    1: add_client,
    2: delete_client,
    3: exit
}

run = True
while run:
    print('''
    1: Add Client
    2: Delete Client
    3: Exit''')
    choice = int(input("Enter your choice (1,2,3): "))
    action = actions.get(choice)
    run = action(stack)
```

Output.

```
    1: Add Client
    2: Delete Client
    3: Exit
Enter your choice (1,2,3): 1
Enter an client name: user1

    1: Add Client
    2: Delete Client
    3: Exit
Enter your choice (1,2,3): 1
Enter an client name: user2
```

```
    1: Add Client
    2: Delete Client
    3: Exit
Enter your choice (1,2,3): 2
Deleted Client:  user2

    1: Add Client
    2: Delete Client
    3: Exit
Enter your choice (1,2,3): 3
Program exited Successfully
```

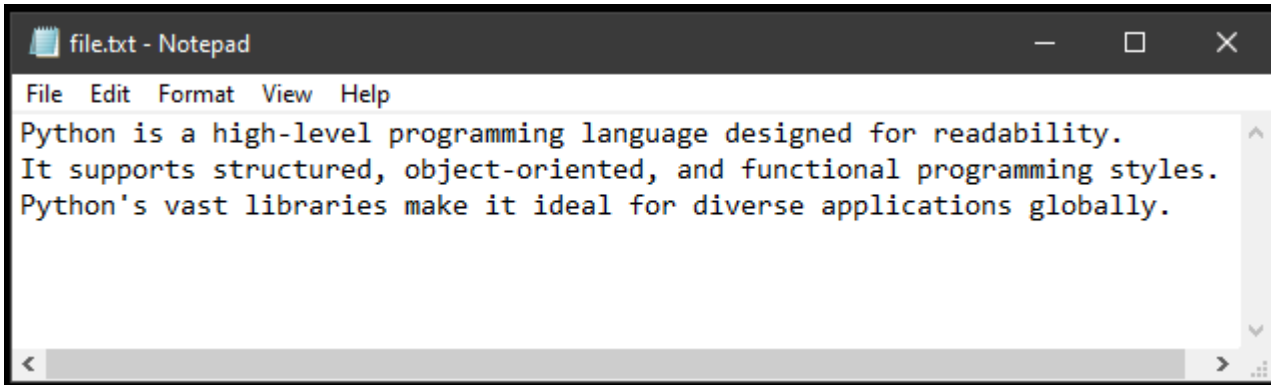
Program 6.

Aim.

Write a program to read a text file by line and display each word separated by a "#"

Program.

```
with open("file.txt", "r") as f:
    for line in f:
        new_line = "#".join(line.split())
        print(new_line + "#")
```



Output.

```
Python#is#a#high-level#programming#language#designed#for#readability.#
It#supports#structured,#object-oriented,#and#functional#programming#styles.#
Python's#vast#libraries#make#it#ideal#for#diverse#applications#globally.#
```

Program 7.

Aim.

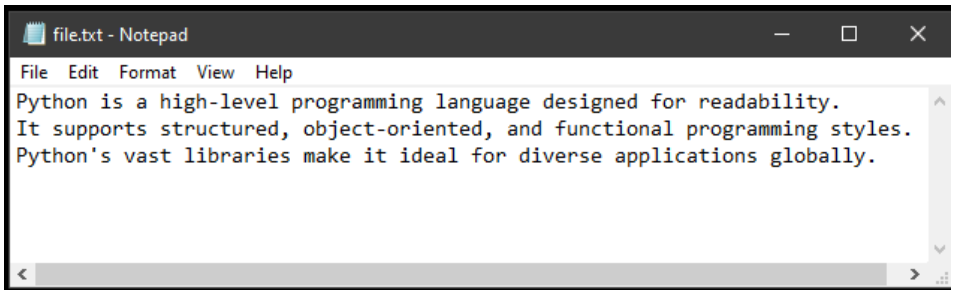
Write a program to read a text file and display the number of vowels, consonants, upper case, lower case characters in file.

Program.

```
with open("file.txt","r") as f:
    text = f.read()
    count_vowels = 0
    count_consonants = 0
    count_upper = 0
    count_lower = 0

    for ch in text:
        if ch.isupper():
            count_upper += 1
        if ch.islower():
            count_lower += 1
        if ch.isalpha():
            if ch in "aeiouAEIOU":
                count_vowels += 1
            else:
                count_consonants += 1

print(f"Vowels: {count_vowels}")
print(f"Consonants: {count_consonants}")
print(f"Uppercase Characters: {count_upper}")
print(f"Lowercase Characters: {count_lower}")
```



Output.

```
Vowels: 66
Consonants: 118
Uppercase Characters: 3
Lowercase Characters: 181
```

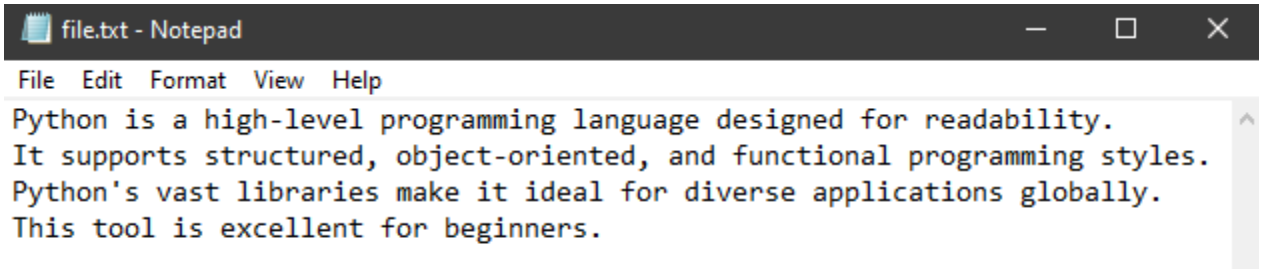
Program 8.

Aim.

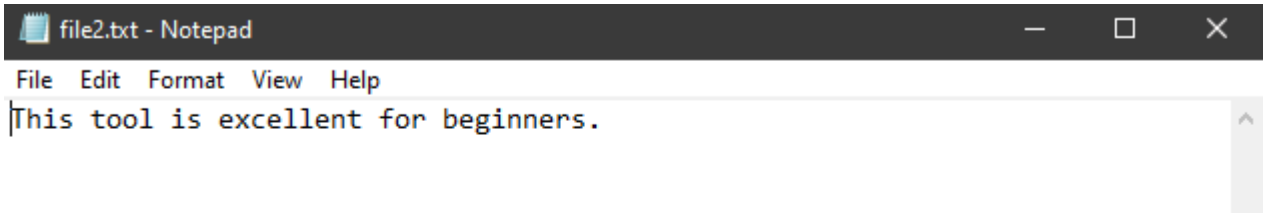
Write a program to remove all the lines that contain the character “a” in a file and write it to another file.

Program.

```
with open('file.txt', 'r') as f:
    with open('file2.txt', 'w') as g:
        data = f.readlines()
        for line in data:
            c_a, c_A = line.count("a"), line.count("A")
            if c_a == 0 and c_A == 0:
                g.write(line)
```



Output.



Program 9.

Aim.

Write a program to create a binary file with name and roll number. Search for a given roll number and display the name, if not found display appropriate message.

Program.

```
import pickle
with open("file.dat", "wb") as f:
    run = True
    while run:
        roll_no = int(input("Enter the Roll No: "))
        name = input("Enter the Name: ")
        pickle.dump( obj: [roll_no, name], f)
        x = int(input("To continue press 1: "))
        if x != 1:
            run = False
```

```
with open("file.dat", "rb") as g:
    details = []
    try:
        while True:
            details.append(pickle.load(g))
    except EOFError:
        found = False
        r_no = int(input("Enter the Roll No to Search: "))
        for line in details:
            if line[0] == r_no:
                print(f"Roll No: {line[0]} , Name: {line[1]}")
                found = True
        if not found:
            print("No record found with the given Roll No.")
```

Output.

```
Enter the Roll No: 1
Enter the Name: Aarav
To continue press 1: 1
Enter the Roll No: 2
Enter the Name: Aditya
To continue press 1: 1
Enter the Roll No: 3
Enter the Name: Dinesh
To continue press 1: 0
Enter the Roll No to Search: 2
Roll No: 2 , Name: Aditya
```

Program 10.

Aim.

Write a program to create a binary file with roll number, name and mark. Input a roll number and update the marks.

Program.

```python
import pickle
with open("file.dat", "wb") as f:
    run = True
    while run:
        roll_no = int(input("Enter the Roll No: "))
        name = input("Enter the Name: ")
        mark = float(input("Enter the Mark: "))
        pickle.dump( obj: [roll_no, name, mark], f)
        x = int(input("To continue press 1: "))
        if x != 1:
            run = False
```

```
with open("file.dat", "rb+") as g:
    details = []
    try:
        while True:
            details.append(pickle.load(g))
    except EOFError:
        pass

    found = False
    r_no = int(input("Enter the Roll No to Search: "))
    for i, line in enumerate(details):
        if line[0] == r_no:
            mrk = float(input("Enter new mark: "))
            details[i][2] = mrk
            found = True
            print("Marks changed")
            print(f"Roll No: {line[0]} , Name: {line[1]}, Marks: {line[2]}")
            break
```

```
    if not found:
        print("No record found with the given Roll No.")
    else:
        g.seek(0)
        g.truncate()
        for record in details:
            pickle.dump(record, g)
```

Output.

```
Enter the Roll No: 1
Enter the Name: Aditi
Enter the Mark: 95
To continue press 1: 1
Enter the Roll No: 2
Enter the Name: Anjali
Enter the Mark: 97
To continue press 1: 1
Enter the Roll No: 3
Enter the Name: Yash
Enter the Mark: 99
To continue press 1: 0
Enter the Roll No to Search: 2
Enter new mark: 98.5
Marks changed
Roll No: 2 , Name: Anjali, Marks: 98.5
```

Program 11.

Aim.

Write a program to create a random number generator that generates random numbers between 1 to 6 simulating a die.

Program.

```
import random
run = True
while run:
    print("="*15, " Welcome to Die Simulator ", "="*15 )
    x = int(input("To Play Press 1: "))
    if x == 1:
        print(f"You got {random.randint( a: 1,  b: 6)}")
    else:
        run = False
    print("="*25, " Thank You ", "="*25, "\n")
```

Output.

```
===============  Welcome to Die Simulator  ===============
To Play Press 1: 1
You got 2
=========================  Thank You  =========================

===============  Welcome to Die Simulator  ===============
To Play Press 1: 0
=========================  Thank You  =========================
```

Source Codes

Program 1:

(

```
def countchar(sentence, character):
```

```
    sentence = sentence.lower()
    character = character.lower()
    count = 0
    for char in sentence:
        if character == char:
            count += 1
        else:
            continue
    if count == 0:
        print(f"{character} is not found")
    else:
        print(f"The no of occurrence of {character} is {count}")
```
)

Program 2:

(
```
def div5():
    list_1, list_2 = [], []
    n = int(input("Enter the no of inputs: "))
    for i in range(1, n+1):
        element = int(input(f"Enter element {i}: "))
        list_1.append(element)
    print(f"List 1: {list_1}")

    for j in list_1:
        if j % 5 == 0:
            list_2.append(j)
    print(f"List of number divisible by 5: {list_2}")
```

)

Program 3:

(
```
def prime(no):
    chk = 0
    for i in range(2, (no//2) + 1):
        if no % i == 0:
            chk += 1
    return chk


while True:
    x = int(input("Enter a number: "))
    is_prime = prime(x)

    if is_prime != 0:
        print("It is not a prime number!")
    else:
        print("It is a prime number")
```
)

Program 4:

(

```
def push(lst):
    x = input("Enter an element: ")
    lst.append(x)
    return True


def pop(lst):
    print("Deleted element: ", lst.pop() if lst else "Stack underflow")
    return True


def peek(lst):
    print("Top element: ", lst[-1] if lst else "Stack underflow")
    return True


def display(lst):
    print("Stack elements: ", lst[::-1] if lst else "Stack underflow")
    return True


def exit(lst):
    print("Program exited Successfully")
    return False


stack = []
actions = {
    1: push,
    2: pop,
    3: display,
    4: peek,
    5: exit
}

run = True
while run:
    print('''
    1: Push
    2: Pop
    3: Display
    4: Peek
    5: Exit''')
    choice = int(input("Enter your choice (1,2,3,4,5): "))
    action = actions.get(choice)
    run = action(stack)
```

)

Program 5:

(

```
def add_client(lst):
    x = input("Enter an client name: ")
    lst.append(x)
    return True


def delete_client(lst):
    print("Deleted Client: ", lst.pop() if lst else "Stack underflow")
    return True


def exit(lst):
    print("Program exited Successfully")
    return False


stack = []
actions = {
    1: add_client,
    2: delete_client,
    3: exit
}

run = True
while run:
    print('''
    1: Add Client
    2: Delete Client
    3: Exit''')
    choice = int(input("Enter your choice (1,2,3): "))
    action = actions.get(choice)
    run = action(stack)
```

)

Program 6:

(

```
with open("file.txt", "r") as f:
    for line in f:
        new_line = "#".join(line.split())
        print(new_line + "#")
```

)

Program 7:

(

```
with open("file.txt","r") as f:
   text = f.read()
   count_vowels = 0
   count_consonants = 0
   count_upper = 0
   count_lower = 0

   for ch in text:
       if ch.isupper():
           count_upper += 1
       if ch.islower():
           count_lower += 1
       if ch.isalpha():
           if ch in "aeiouAEIOU":
               count_vowels += 1
           else:
               count_consonants += 1

print(f"Vowels: {count_vowels}")
print(f"Consonants: {count_consonants}")
print(f"Uppercase Characters: {count_upper}")
print(f"Lowercase Characters: {count_lower}")
```

)

Program 8:

(

```
with open('file.txt', 'r') as f:
   with open('file2.txt', 'w') as g:
       data = f.readlines()
       for line in data:
           c_a, c_A = line.count("a"), line.count("A")
           if c_a == 0 and c_A == 0:
               g.write(line)
```

)

Program 9:

(

```
import pickle
with open("file.dat", "wb") as f:
   run = True
   while run:
       roll_no = int(input("Enter the Roll No: "))
       name = input("Enter the Name: ")
```

```
        pickle.dump([roll_no, name], f)
        x = int(input("To continue press 1: "))
        if x != 1:
            run = False

with open("file.dat", "rb") as g:
    details = []
    try:
        while True:
            details.append(pickle.load(g))
    except EOFError:
        found = False
        r_no = int(input("Enter the Roll No to Search: "))
        for line in details:
            if line[0] == r_no:
                print(f"Roll No: {line[0]} , Name: {line[1]}")
                found = True
        if not found:
            print("No record found with the given Roll No.")
```

)

Program 10:

(

```
import pickle
with open("file.dat", "wb") as f:
    run = True
    while run:
        roll_no = int(input("Enter the Roll No: "))
        name = input("Enter the Name: ")
        mark = float(input("Enter the Mark: "))
        pickle.dump([roll_no, name, mark], f)
        x = int(input("To continue press 1: "))
        if x != 1:
            run = False

with open("file.dat", "rb+") as g:
    details = []
    try:
        while True:
            details.append(pickle.load(g))
    except EOFError:
        pass

    found = False
    r_no = int(input("Enter the Roll No to Search: "))
    for i, line in enumerate(details):
        if line[0] == r_no:
            mrk = float(input("Enter new mark: "))
            details[i][2] = mrk
            found = True
```

```
            print("Marks changed")
            print(f"Roll No: {line[0]} , Name: {line[1]}, Marks: {line[2]}")
            break

    if not found:
        print("No record found with the given Roll No.")
    else:
        g.seek(0)
        g.truncate()
        for record in details:
            pickle.dump(record, g)
```
)

Program 11:

(
```
import random
run = True
while run:
    print("="*15, " Welcome to Die Simulator ", "="*15 )
    x = int(input("To Play Press 1: "))
    if x == 1:
        print(f"You got {random.randint(1, 6)}")
    else:
        run = False
    print("="*25, " Thank You ", "="*25, "\n")
```

)