Sardar Vallabhbhai National Institute of Technology

Surat-395007

Web Programming and Python (AI104)

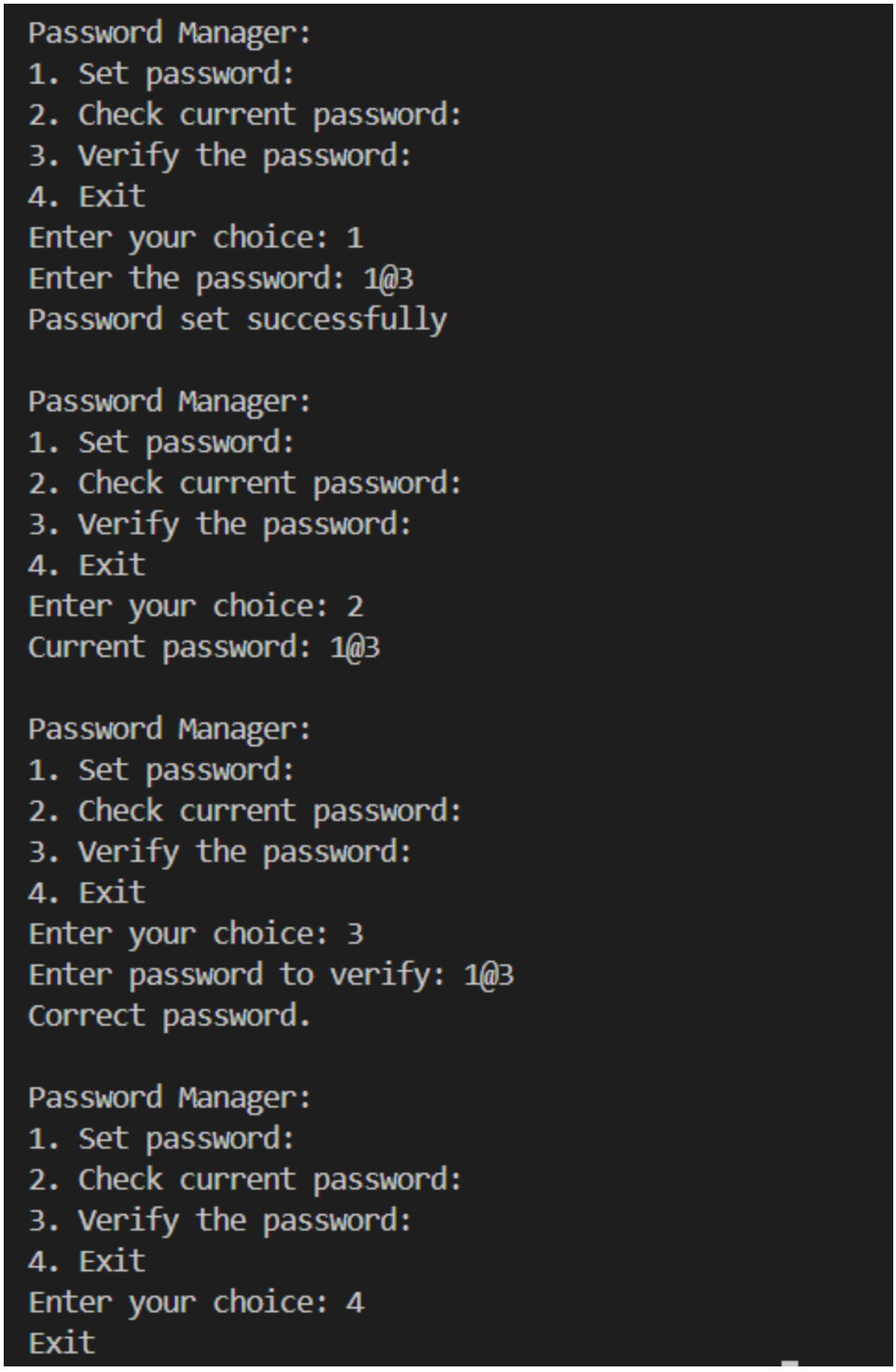
Assignment – 6

Object-Oriented Programming
ROLL NO: I24AI001

1. Write a class called Password_manager. The class should have a list called old_passwords that
holds all of the user's past passwords. The last item of the list is the user's current password. There should be a method called get_password that returns the current password and a method called set_password that sets the user's password. The set_password method should only change the password if the attempted password is different from all the user's past passwords. Finally, create a method called is_correct that receives a string and returns a boolean True or False depending on whether the string is equal to the current password or not.

```
class password_manager():
    def __init__(self):
        self.old_passwords=[]

    def get_password(self):
        if self.old_passwords:
            return self.old_passwords[-1]
        return None

    def set_password(self, new_password):
        if new_password not in self.old_passwords:
            self.old_passwords.append(new_password)
            print("Password set successfully")
        else:
            print("New password not the same")

    def is_correct(self,password):
        return password==self.get_password()

def main():
    pm = password_manager()
```

```
    while True:
        print("\nPassword Manager: ")
        print("1. Set password: ")
        print("2. Check current password: ")
        print("3. Verify the password: ")
        print("4. Exit")
        choice=input("Enter your choice: ")

        if choice=='1':
            new_password=input("Enter the password: ")
            pm.set_password(new_password)

        elif choice=='2':
            current_password=pm.get_password()
            if current_password:
                    print(f"Current password: {current_password}")
            else:
                print("No password")

        elif choice == '3':
            password = input("Enter password to verify: ")
            if pm.is_correct(password):
                print("Correct password.")
            else:
                print("Incorrect password.")

        elif choice == '4':
            print("Exit")
            break

        else:
            print("Invalid choice")

if __name__=="__main__":
    main()
```



2. Write a class called Rock_paper_scissors that implements the logic of the game Rock paper-scissors. For this game the user plays against the computer for a certain number of rounds.

Your class should have fields for the how many rounds there will be, the current round number,

and the number of wins each player has. There should be methods for getting the computer's choice, finding the winner of a round, and checking to see if someone has one the (entire) game. You may want more methods.

```
import random

class Rock_paper_scissors():
    def __init__(self,rounds):
        self.rounds=rounds
        self.current_round=1
        self.player_wins=0
        self.computer_wins=0
        self.choices=['rock','paper','scissors']

    def get_computer_choice(self):
        return random.choice(self.choices)

    def find_winner(self,player_choice, computer_choice):
        if player_choice==computer_choice:
            return "draw"
        elif (player_choice == 'rock' and computer_choice == 'scissors')
or \
             (player_choice == 'scissors' and computer_choice == 'paper')
or \
             (player_choice == 'paper' and computer_choice == 'rock'):
            return "player"
        else:
            return "computer"

    def play_round(self, player_choice):
        if player_choice not in self.choices:
            return "Invalid choice"

        computer_choice=self.get_computer_choice()
        print(f"Computer choice: {computer_choice}")

        winner=self.find_winner(player_choice, computer_choice)

        if winner=='player':
            self.player_wins+=1
```

```
            print("You won!")
        elif winner=='computer':
            self.computer_wins+=1
            print('Computer won!')
        else:
            print("Draw")

        self.current_round+=1

    def won_game(self):
        if self.current_round>self.rounds:
            if self.player_wins>self.computer_wins:
                print(f'\nYou won:
{self.player_wins}-{self.computer_wins}')
                return True
            elif self.computer_wins>self.player_wins:
                print(f"Computer won:
{self.computer_wins}-{self.player_wins}")
                return True
            else:
                print(f'Draw: {self.player_wins}-{self.computer_wins}')
                return True
        return False

def main():
    rounds=int(input("Enter no of rounds: "))
    game=Rock_paper_scissors(rounds)

    while game.current_round<=game.rounds:
        print(f"\nRound: {game.current_round} of {game.rounds}")
        player_choice=input('Enter your choice: ')
        game.play_round(player_choice)

        if game.won_game():
            break

if __name__=="__main__":
    main()
```

```
Enter no of rounds: 2

Round: 1 of 2
Enter your choice: rock
Computer choice: paper
Computer won!

Round: 2 of 2
Enter your choice: paper
Computer choice: rock
You won!
Draw: 1-1
```

3. Write a class called Converter. The user will pass a length and a unit when declaring an object
from the class—for example, c = Converter(9,'inches'). The possible units are inches, feet, yards, miles, kilometers, meters, centimeters, and millimeters. For each of these units there should be a method that returns the length converted into those units. For example, using the Converter object created above, the user could call c.feet() and should get 0.75 as the result.

```
class Converter():
    conversion_factors={
        'inches': 1,
        'feet': 1 / 12,
        'yards': 1 / 36,
        'miles': 1 / 63360,
        'kilometers': 1 / 39370.1,
        'meters': 1 / 39.3701,
        'centimeters': 2.54,
        'millimeters': 25.4}

    def __init__(self, length, unit):
        self.length=length
        self.unit=unit.lower()

        if self.unit not in Converter.conversion_factors:
            raise ValueError(f"Invalid unit:
{','.join(Converter.conversion_factors.keys())}")

    def convert_to_inches(self):
```

```
        return self.length*Converter.conversion_factors[self.unit]

    def inches(self):
        return self.convert_to_inches()

    def feet(self):
        return
self.convert_to_inches()/Converter.conversion_factors['feet']

    def yards(self):
        return
self.convert_to_inches()/Converter.conversion_factors['yards']

    def miles(self):
        return
self.convert_to_inches()/Converter.conversion_factors['miles']

    def kilometers(self):
        return
self.convert_to_inches()/Converter.conversion_factors['kilometers']

    def meters(self):
        return
self.convert_to_inches()/Converter.conversion_factors['meters']

    def centimeters(self):
        return
self.convert_to_inches()/Converter.conversion_factors['centimeters']

    def millimeters(self):
        return
self.convert_to_inches()/Converter.conversion_factors['millimeters']

def main():
    length=float(input("Enter the length: "))
    unit=input("Enter the unit: ")
    c=Converter(length, unit)

    print(f"{length} {unit} is equivalent to:")
    print(f"{c.inches()} inches")
```

```
    print(f"{c.feet()} feet")
    print(f"{c.yards()} yards")
    print(f"{c.miles()} miles")
    print(f"{c.kilometers()} kilometers")
    print(f"{c.meters()} meters")
    print(f"{c.centimeters()} centimeters")
    print(f"{c.millimeters()} millimeters")

if __name__=="__main__":
    main()
```

```
Enter the length: 4
Enter the unit: meters
4.0 meters is equivalent to:
0.10159994513602963 inches
1.2191993416323557 feet
3.6575980248970668 yards
6437.372523818837 miles
4000.0 kilometers
4.0 meters
0.039999997840001166 centimeters
0.0039999997840001167 millimeters
```