**Task – 01**

**Build a Temperature Conversion Program**

```
def celsius_to_fahrenheit(celsius):
return (celsius * 9/5) + 32

def celsius_to_kelvin(celsius):
return celsius + 273.15

def fahrenheit_to_celsius(fahrenheit):
return (fahrenheit - 32) * 5/9

def fahrenheit_to_kelvin(fahrenheit):
return (fahrenheit + 459.67) * 5/9

def kelvin_to_celsius(kelvin):
return kelvin - 273.15

def kelvin_to_fahrenheit(kelvin):
return kelvin * 9/5 - 459.67

def main():
print("Welcome to the Temperature Conversion Program")
temperature = float(input("Enter the temperature value: "))
unit = input("Enter the unit of measurement (Celsius, Fahrenheit, or Kelvin): ").lower()

if unit == "celsius":
fahrenheit = celsius_to_fahrenheit(temperature)
```

```
kelvin = celsius_to_kelvin(temperature)

print(f"{temperature} degrees Celsius is equal to {fahrenheit:.2f} degrees Fahrenheit and {kelvin:.2f} Kelvin.")

elif unit == "fahrenheit":

celsius = fahrenheit_to_celsius(temperature)

kelvin = fahrenheit_to_kelvin(temperature)

print(f"{temperature} degrees Fahrenheit is equal to {celsius:.2f} degrees Celsius and {kelvin:.2f} Kelvin.")

elif unit == "kelvin":

celsius = kelvin_to_celsius(temperature)

fahrenheit = kelvin_to_fahrenheit(temperature)

print(f"{temperature} Kelvin is equal to {celsius:.2f} degrees Celsius and {fahrenheit:.2f} Fahrenheit.")

else:

print("Invalid unit of measurement. Please enter Celsius, Fahrenheit, or Kelvin.")


if __name__ == "__main__":

main()
```

**OUTPUT**

```
Output                                                    Clear
Welcome to the Temperature Conversion Program
Enter the temperature value: 35
Enter the unit of measurement (Celsius, Fahrenheit, or Kelvin): Celsius
35.0 degrees Celsius is equal to 95.00 degrees Fahrenheit and 308.15 Kelvin.

=== Code Execution Successful ===
```

**Task – 02**

**Create a Guessing Game**

```
import random
def guessing_game():
print("Welcome to the Guessing Game!")
print("I have selected a random number between 1 and 100.")
print("Try to guess it!")

secret_number = random.randint(1, 100)
attempts = 0

while True:
guess = int(input("Enter your guess: "))
attempts += 1

if guess < secret_number:
print("Too low! Try again.")
elif guess > secret_number:
print("Too high! Try again.")
else:
print(f"Congratulations! You've guessed the correct number {secret_number} in {attempts} attempts.")
break

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

**OUTPUT**

```
Output                                                    Clear
Welcome to the Guessing Game!
I have selected a random number between 1 and 100.
Try to guess it!
Enter your guess: 40
Too high! Try again.
Enter your guess: 25
Too high! Try again.
Enter your guess: 15
Too low! Try again.
Enter your guess: 20
Too high! Try again.
Enter your guess: 22
Too high! Try again.
Enter your guess: 16
Too low! Try again.
Enter your guess: 18
Congratulations! You've guessed the correct number 18 in 7 attempts.

=== Code Execution Successful ===
```

**Task – 03**

**Implement a simple contact management system**

```
import json

# Function to load contacts from a file
def load_contacts():
try:
with open("contacts.json", "r") as file:
return json.load(file)
except FileNotFoundError:
return {}
```

```
# Function to save contacts to a file
def save_contacts(contacts):
with open("contacts.json", "w") as file:
json.dump(contacts, file)

# Function to add a new contact
def add_contact(contacts):
name = input("Enter contact's name: ")
phone = input("Enter contact's phone number: ")
email = input("Enter contact's email address: ")
contacts[name] = {"phone": phone, "email": email}
print("Contact added successfully!")

# Function to view all contacts
def view_contacts(contacts):
if contacts:
print("List of Contacts:")
for name, info in contacts.items():
print(f"Name: {name}, Phone: {info['phone']}, Email: {info['email']}")
else:
print("No contacts found.")

# Function to edit a contact
def edit_contact(contacts):
name = input("Enter the name of the contact you want to edit: ")
if name in contacts:
print(f"Editing contact: {name}")
```

```
phone = input("Enter new phone number (press Enter to keep existing): ")
email = input("Enter new email address (press Enter to keep existing): ")
if phone:
contacts[name]["phone"] = phone
if email:
contacts[name]["email"] = email
print("Contact updated successfully!")
else:
print("Contact not found.")

# Function to delete a contact
def delete_contact(contacts):
name = input("Enter the name of the contact you want to delete: ")
if name in contacts:
del contacts[name]
print("Contact deleted successfully!")
else:
print("Contact not found.")

# Main function
def main():
contacts = load_contacts()
while True:
print("\nWelcome to the Simple Contact Management System")
print("1. Add Contact")
print("2. View Contacts")
print("3. Edit Contact")
```

```
print("4. Delete Contact")
print("5. Exit")
choice = input("Enter your choice: ")

if choice == "1":
add_contact(contacts)
elif choice == "2":
view_contacts(contacts)
elif choice == "3":
edit_contact(contacts)
elif choice == "4":
delete_contact(contacts)
elif choice == "5":
save_contacts(contacts)
print("Thank you for using the Contact Management System. Goodbye!")
break
else:
print("Invalid choice. Please enter a number from 1 to 5.")

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

**OUTPUT**

Output Clear

```
Welcome to the Simple Contact Management System
1. Add Contact
2. View Contacts
3. Edit Contact
4. Delete Contact
5. Exit
Enter your choice: 1
Enter contact's name: Isabella Gracia
Enter contact's phone number: +202 5783128234
Enter contact's email address: isabella1122@gmail.com
Contact added successfully!

Welcome to the Simple Contact Management System
1. Add Contact
2. View Contacts
3. Edit Contact
4. Delete Contact
5. Exit
Enter your choice: 2
List of Contacts:
Name: Isabella Gracia, Phone: +202 5783128234, Email: isabella1122@gmail.com

Welcome to the Simple Contact Management System
1. Add Contact
2. View Contacts
3. Edit Contact
4. Delete Contact
5. Exit
Enter your choice: 3
Enter the name of the contact you want to edit: Isabella Gracia
Editing contact: Isabella Gracia
Enter new phone number (press Enter to keep existing): +202 5878987091
Enter new email address (press Enter to keep existing): isabella@gmail.com
```

```
Contact updated successfully!

Welcome to the Simple Contact Management System
1. Add Contact
2. View Contacts
3. Edit Contact
4. Delete Contact
5. Exit
Enter your choice: 4
Enter the name of the contact you want to delete: Isabella Gracia
Contact deleted successfully!

Welcome to the Simple Contact Management System
1. Add Contact
2. View Contacts
3. Edit Contact
4. Delete Contact
5. Exit
```

```
Enter your choice: 5
Thank you for using the Contact Management System. Goodbye!
```

**Task – 04**

**Implement a Suduko Solver**

```
def is_valid_move(board, row, col, num):
# Check if the number is already present in the row
if num in board[row]:
return False

# Check if the number is already present in the column
if num in [board[i][col] for i in range(9)]:
return False

# Check if the number is already present in the 3x3 subgrid
start_row, start_col = 3 * (row // 3), 3 * (col // 3)
for i in range(start_row, start_row + 3):
for j in range(start_col, start_col + 3):
if board[i][j] == num:
return False

return True

def solve_sudoku(board):
empty_cell = find_empty_cell(board)
if not empty_cell:
return True  # Puzzle solved

row, col = empty_cell
for num in range(1, 10):
```

```
if is_valid_move(board, row, col, num):
board[row][col] = num
if solve_sudoku(board):
return True
board[row][col] = 0  # Backtrack
return False

def find_empty_cell(board):
for i in range(9):
for j in range(9):
if board[i][j] == 0:
return (i, j)
return None

def print_board(board):
for row in board:
print(" ".join(map(str, row)))

# Example puzzle (0 represents empty cells)
puzzle = [
[5, 3, 0, 0, 7, 0, 0, 0, 0],
[6, 0, 0, 1, 9, 5, 0, 0, 0],
[0, 9, 8, 0, 0, 0, 0, 6, 0],
[8, 0, 0, 0, 6, 0, 0, 0, 3],
[4, 0, 0, 8, 0, 3, 0, 0, 1],
[7, 0, 0, 0, 2, 0, 0, 0, 6],
[0, 6, 0, 0, 0, 0, 2, 8, 0],
```

```
[0, 0, 0, 4, 1, 9, 0, 0, 5],
[0, 0, 0, 0, 8, 0, 0, 7, 9]
]

if solve_sudoku(puzzle):
print("Sudoku puzzle solved:")
print_board(puzzle)
else:
print("No solution exists for the given puzzle.")
```

**OUTPUT**

```
Output                                                    Clear
Sudoku puzzle solved:
5 3 4 6 7 8 9 1 2
6 7 2 1 9 5 3 4 8
1 9 8 3 4 2 5 6 7
8 5 9 7 6 1 4 2 3
4 2 6 8 5 3 7 9 1
7 1 3 9 2 4 8 5 6
9 6 1 5 3 7 2 8 4
2 8 7 4 1 9 6 3 5
3 4 5 2 8 6 1 7 9

=== Code Execution Successful ===
```

**Task – 05**

**Web Scraping**

```
import requests
from bs4 import BeautifulSoup
import csv
```

```
def scrape_product_info(url):
    headers = {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3'}
    response = requests.get(url, headers=headers)

    if response.status_code == 200:
        soup = BeautifulSoup(response.content, 'html.parser')
        products = []

        # Find product information
        for product in soup.find_all('div', class_='product'):
            name = product.find('h2', class_='product-name').text.strip()
            price = product.find('span', class_='product-price').text.strip()
            rating = product.find('div', class_='product-rating').text.strip()

            products.append({
                'Name': name,
                'Price': price,
                'Rating': rating
            })

        return products
    else:
        print("Failed to retrieve data.")
        return None
```

```
def save_to_csv(data, filename):
    with open(filename, 'w', newline='', encoding='utf-8') as csvfile:
        fieldnames = ['Name', 'Price', 'Rating']
        writer = csv.DictWriter(csvfile, fieldnames=fieldnames)

        writer.writeheader()
        for product in data:
            writer.writerow(product)

if __name__ == "__main__":
    url = 'https://example.com/products'  # Replace this with the URL of the e-commerce website
    products = scrape_product_info(url)
    if products:
        save_to_csv(products, 'products.csv')
        print("Product information extracted and saved to products.csv.")
```

**OUTPUT**

```
Name,Price,Rating
Product 1,19.99,4.5
Product 2,29.99,3.8
Product 3,14.99,4.2
```