Netaji Subash University of Technology

Practical File

On

Computer Programming

Submitted To:

Anamika Rajput

Submitted By:

Roshan Sharma

2024UCA1908

CSAI - 2

INDEX

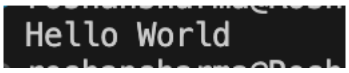
Program-1

1(a): Write a Python program to print "Hello, World!"

Code:

```
print("Hello, World!")
```

Output:



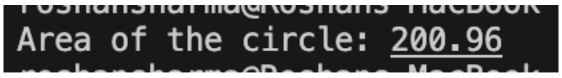
```
Hello World
```

1(b): Write a Python program to calculate the area of a circle given the radius.

Code:

```
def area_of_circle(radius):
        return 3.14 * (radius ** 2)
radius = 8
print(f"Area of the circle: {area_of_circle(radius)}")
```

Output:



```
Area of the circle: 200.96
```

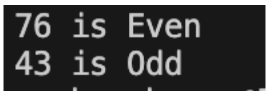
Program-2

2(a): Write a Python program to check if a number is even or odd.

Code:

```
def check_even_odd(num):
    if num % 2 == 0:
        return "Even"
    else:
        return "Odd"
num1 = 76
num2 = 43
print(f" {num1} is {check_even_odd(num1)}")
print(f" {num2} is {check_even_odd(num2)}")
```

Output:



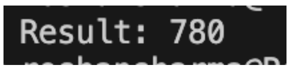
```
76 is Even
43 is Odd
```

2(b): Implement a simple calculator using conditional statements.

Code:

```
def calculator (a, b, operation):
    if operation == "add":
        return a + b
    elif operation == "subtract":
        return a - b
    elif operation == "multiply":
        return a * b
    elif operation == "divide":
        return a / b
print(f"Result: {calculator(12, 65, 'multiply')}")
```

Output:



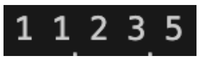
```
Result: 780
```

2(c): Write a Python program to print the Fibonacci series using a for loop.

Code:

```
def fibonacci(n):
   prev1, prev2 = 0,1
   for i in range(2, n+1):
      curr = prev1 +prev2
      prev2 = prev1
      prev1 = curr
      print(prev1, end=" ")
fibonacci(6)
```

Output:



```
1 1 2 3 5
```

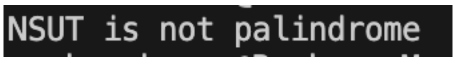
Program-3

3(a): Implement a function to check if a given string is a palindrome.

Code:

```
def is_palindrome(s):
    if s==s[::-1]:
        return True
string = "NSUT"
if is_palindrome(string):
    print(f"{string} is palindrome")
else:
    print(f"{string} is not palindrome")
```

Output:



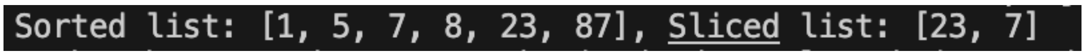
```
NSUT is not palindrome
```

3(b): Perform various operations on lists (e.g., sorting, slicing).

Code:

```
my_list = [5,23,7,1,87,8]
sorted_list = sorted(my_list)
sliced_list = my_list [1:3]
print (f"Sorted list: {sorted_list}, Sliced list: {sliced_list}")
```

Output:



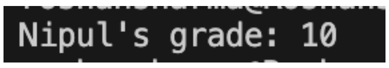
```
Sorted list: [1, 5, 7, 8, 23, 87], Sliced list: [23, 7]
```

3(c): Use dictionaries to store and retrieve student grades.

Code:

```
grades = {"Prankush": 75, "Nipul": 10, "Snehil": 54}

print (f"Nipul's grade: {grades ['Nipul']}")
```

Output:



```
Nipul's grade: 10
```

Program-4

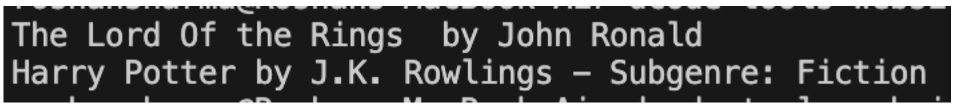
4(a): Create a class to represent a book with attributes and methods.

4(b): Implement inheritance by creating subclasses for different types of books.

Code:

```
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author
    def get_info(self):
        return f"{self.title} by {self.author}"
class Fiction(Book):
    def __init__(self, title, author, subgenre="Fiction"):
        super().__init__(title, author)
        self.subgenre = subgenre
    def get_info(self):
        return super().get_info() + f" - Subgenre: {self.subgenre}"
b1=Book("The Lord Of the Rings ", "John Ronald")
fb1=Fiction("Harry Potter", "J.K. Rowlings")
print(b1.get_info())
print(fb1.get_info())
```

Output:



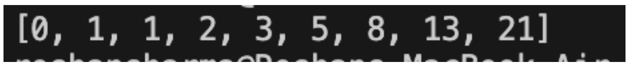
```
The Lord Of the Rings  by John Ronald
Harry Potter by J.K. Rowlings - Subgenre: Fiction
```

4(c): Write a generator function to generate the Fibonacci series.

Code:

```
def fibonacci_gen(n):
    a, b = 0, 1
    while n>0:
        yield a
        c=a+b
        a=b
        b=c
        n-=1
fibo=fibonacci_gen(9)
print(list(fibo))
```

Output:



```
[0, 1, 1, 2, 3, 5, 8, 13, 21]
```

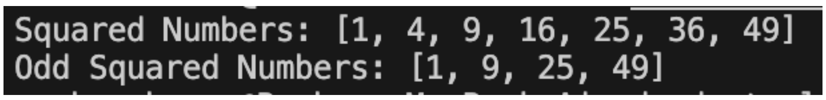
Program-5

5(a): Use lambda functions, map, and filter to perform operations on a list.

Code:

```
numbers = [1, 2, 3, 4, 5, 6, 7]

squared_numbers = list(map(lambda x: x ** 2, numbers))

print("Squared Numbers:", squared_numbers)

odd_squared_numbers = list(filter(lambda x: x % 2 != 0, squared_numbers))

print("Odd Squared Numbers:", odd_squared_numbers)
```

Output:



```
Squared Numbers: [1, 4, 9, 16, 25, 36, 49]
Odd Squared Numbers: [1, 9, 25, 49]
```

5(b): Create a module that contains functions for mathematical operations.

Code:

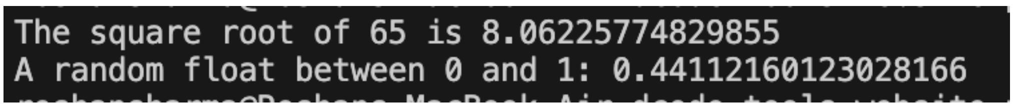
```
def add(a, b):

    return a + b

def subtract(a, b):

    return a - b

def multiply(a, b):

    return a * b

def divide(a, b):

    if b == 0:

        raise ValueError("Cannot divide by zero.")

    return a / b

def power(base, power):

    return base ** power
```

5(c): Import and use functions from external packages (e.g., math, random).

Code:

```
import math

import random

num = 65

square_root = math.sqrt(num)

print(f"The square root of {num} is {square_root}")

random_float = random.random()

print(f"A random float between 0 and 1: {random_float}")
```

Output:



```
The square root of 65 is 8.06225774829855
A random float between 0 and 1: 0.44112160123028166
```

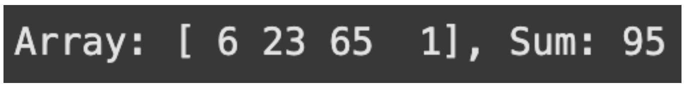
Program-6

6(a): Create and manipulate NumPy arrays.

Code:

```
import numpy as np

arr = np.array([6,23,65,1])

print(f"Array: {arr}, Sum: {np. sum(arr)}")
```

Output:



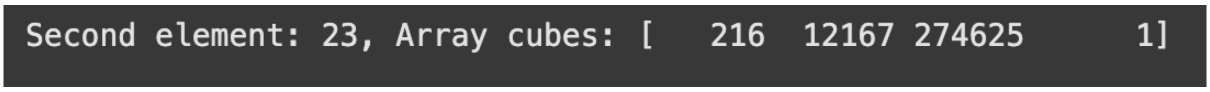
```
Array: [ 6 23 65  1], Sum: 95
```

6(b): Perform basic operations and indexing on arrays.

Code:

```
import numpy as np

arr = np.array([6,23,65,1])

print(f"Second element: {arr[1]}, Array cubes: {arr ** 3}")
```

Output:



```
Second element: 23, Array cubes: [   216  12167 274625      1]
```

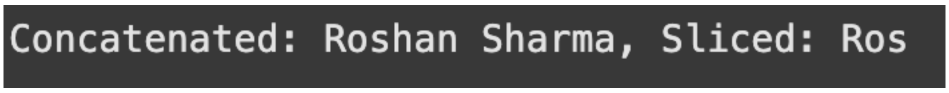
Program-7

7(a): Implement string operations (e.g., concatenation, slicing).

Code:

```
str1 = "Roshan"

str2 = "Sharma"

concat_str = str1 + " " + str2

sliced_str = concat_str[0:3]

print (f"Concatenated: {concat_str}, Sliced: {sliced_str}")
```

Output:



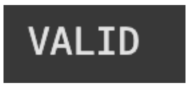
```
Concatenated: Roshan Sharma, Sliced: Ros
```

7(b): Use regular expressions to validate email addresses.

Code:

```
import re

def validate_email(email):

    pattern = r'^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$'

    if re.match(pattern, email):

        return True

    else:

        return False

em=“roshan12@gmail.com”

if validate_email(em):

    print("VALID")

else:

    print("NOT VALID")
```

Output:



```
VALID
```

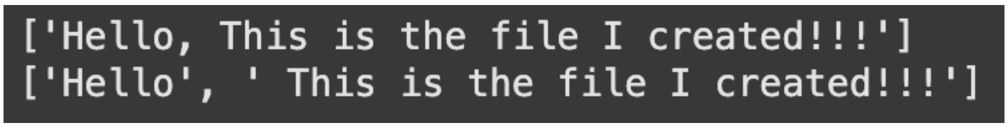
Program-8

8(a): Read data from a text file and perform operations.

Code:

```
with open('file.txt', 'r') as f:
    data = f.readlines()
    print(data)
    m_data = data[0].split(",")
    print(m_data)
```

Output:



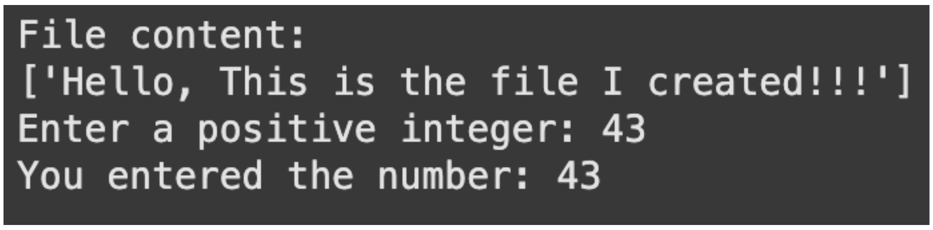
```
['Hello, This is the file I created!!!']
['Hello', ' This is the file I created!!!']
```

8(b): Handle exceptions for file operations and input validation.

Code:

```
try:
    with open('file.txt', 'r') as f:
        data = f.readlines()
        print("File content:")
        print(data)
except:
    print("Error!!!")
def get_positive_integer():
    while True:
        try:
            num = int(input("Enter a positive integer: "))
            if num <= 0:
                raise ValueError("The number must be positive.")
            return num
    except ValueError as err:
        print(f"Invalid input: {err}. Please try again.")
p_int= get_positive_integer()
print(f"You entered the number: {p_int}")
```

Output:



```
File content:
['Hello, This is the file I created!!!']
Enter a positive integer: 43
You entered the number: 43
```