IT1160 - Discrete Mathematics

Lab Sheet 09

Part A

01) Write a recursive function factorial(n) that returns the factorial of a given positive integer n

Ex- **factorial(5) → 120**

02) Write a recursive function sum_of_digits(n) that returns the sum of digits of a non-negative integer n.

Ex- **sum_of_digits(1234) → 10 # (1 + 2 + 3 + 4)**

03) Write a recursive function is_palindrome(s) that checks whether a string s is a palindrome (same forwards and backwards). Ignore case.

Ex-

is_palindrome("Madam") → True

is_palindrome("Hello") → False

04) Write a recursive function count_vowels(s) that returns the number of vowels (a, e, i, o, u) in a string s.

Ex- **count_vowels("recursion") → 4**

05) Write a recursive function reverse_list(lst) that returns the reverse of a list without using slicing or loops.

Ex- **reverse_list([1, 2, 3, 4]) → [4, 3, 2, 1]**

06) Analyze the Time Complexity of following programs.

(i)

```
def fun1(arr):
    for i in arr:
        for j in arr:
            print(i, j)
```

(ii)

```
def fun2(n):
    if n == 0:
        return 0
    return n + fun2(n - 1)
```

(iii)

```
def fun3(arr, key):
    for i in arr:
        if i == key:
            return True
    return False
```

(iv)

```python
def fun4(arr, key):
    low = 0
    high = len(arr) - 1
    while low <= high:
        mid = (low + high) // 2
        if arr[mid] == key:
            return True
        elif arr[mid] < key:
            low = mid + 1
        else:
            high = mid - 1
    return False
```

(v)

```python
def fun5(arr):
    result = []
    for item in arr:
        result.append(item * 2)
    return result
```

Part B

01) Write a recursive function fibonacci(n) that returns the nth Fibonacci number (0-indexed).

02) Write a recursive function reverse_string(s) that returns the reverse of the string s.

03) Write a recursive function count_char(s, ch) that counts how many times the character ch appears in the string s.

04) Write a recursive function sum_list(lst) that returns the sum of all elements in a list.

05) Write a recursive function contains(lst, x) that checks whether an element x exists in the list lst.

06) Write a recursive function count_even(lst) that returns the number of even integers in the list lst.

07) Write a recursive function gcd(a, b) that returns the greatest common divisor (GCD) of two integers using Euclid's algorithm.

08) Write a recursive function remove_spaces(s) that returns the string s without any spaces.

09) Write a recursive function sum_nested(lst) that returns the sum of all integers in a list, including any nested lists.

10) Write a recursive function find_min(lst) that returns the smallest number in a non-empty list lst.

Object Oriented Programming in Python

Class and Object Basics
A class is a blueprint for creating objects. An object is an instance of a class.

Example:

```
class Student:
    pass
s1 = Student()  # s1 is an object of class Student
```

Instance Attributes
Attributes that are unique to each object and defined inside the constructor.

Example:

```
class Student:
    def __init__(self, name, age):
        self.name = name
```

```
        self.age = age

s1 = Student("Alice", 20)
print(s1.name)  # Output: Alice
```

Instance Methods
Functions defined in a class that operate on object attributes.

Example:

```
class Student:
    def __init__(self, name):
        self.name = name

    def greet(self):
        return f"Hello, my name is {self.name}"

s1 = Student("Bob")
print(s1.greet())  # Output: Hello, my name is Bob
```

Class Attributes
Attributes shared among all instances of a class.

Example:

```
class Student:
    school = "ABC School"  # Class attribute

    def __init__(self, name):
        self.name = name

s1 = Student("Charlie")
```

```
s2 = Student("David")


print(s1.school)  # Output: ABC School
print(s2.school)  # Output: ABC School
```

Modifying Class Attributes
You can modify class attributes using the class name.

Example:

```
Student.school = "XYZ School"
print(s1.school)  # Output: XYZ School
```

Accessing vs. Overriding Class Attributes
If you assign a value to a class attribute using an object, it creates an instance attribute.

Example:

```
s1.school = "New School"  # Creates a new instance attribute for s1
print(s1.school)  # Output: New School
print(s2.school)  # Output: XYZ School
```

Answers of Part A

Question 01

```
def factorial(n):
    if n == 0 or n == 1:
        return 1
    return n * factorial(n - 1)

print(factorial(5))
```

```
120
```

Question 02

```
def sum_of_digits(n):
    if n == 0:
        return 0
    return n % 10 + sum_of_digits(n // 10)

print(sum_of_digits(1234))
```

```
10
```

Question 03

```
def is_palindrome(s):
    s = s.lower()
    if len(s) <= 1:
        return True
    if s[0] != s[-1]:
        return False
    return is_palindrome(s[1:-1])

print(is_palindrome("Madam"))
print(is_palindrome("Adam"))
```

```
True
False
```

Question 04

```
def count_vowels(s):
    vowels = 'aeiouAEIOU'
    if s == "":
        return 0
    return (s[0] in vowels) + count_vowels(s[1:])

print(count_vowels("apple"))
```

```
2
```

Question 05

```python
def reverse_list(lst):
    if len(lst) == 0:
        return []
    return [lst[-1]] + reverse_list(lst[:-1])

print(reverse_list([1, 2, 3, 4]))
```

```
[4, 3, 2, 1]
```

Question 06 (i)

```python
def fun1(arr):
    for i in arr:
        for j in arr:
            print(i, j)
```

$O(n^2)$

Question 06 (ii)

```python
def fun2(n):
    if n == 0:
        return 0
    return n + fun2(n - 1)
```

O(n)

Question 06 (iii)

```python
def fun3(arr, key):
    for i in arr:
        if i == key:
            return True
    return False
```

O(n)

Question 06 (iv)

```python
def fun4(arr, key):
    low = 0
    high = len(arr) - 1
    while low <= high:
        mid = (low + high) // 2
        if arr[mid] == key:
            return True
        elif arr[mid] < key:
            low = mid + 1
        else:
            high = mid - 1
    return False
```

O(log n)

Question 6 (v)

```
def fun5(arr):
    result = []
    for item in arr:
        result.append(item * 2)
    return result
```

O(n)