

LAB-2

DATE:05/06/2024

1. Write a program to find the reverse of a given number using recursive.

CODING:

```
2 usages
1  def reverse_number(number, reverse=0):
2      if number == 0:
3          return reverse
4      else:
5          digit = number % 10
6          reverse = reverse * 10 + digit
7          return reverse_number(number // 10, reverse)
8
9  num = 12345
10 result = reverse_number(num)
11 print(f"Original Number: {num}")
12 print(f"Reversed Number: {result}")
```

OUTPUT:

```
C:\Users\vinot\PycharmProjects\pythonProje
Original Number: 12345
Reversed Number: 54321

Process finished with exit code 0
```

2. Write a program to find the perfect number.

CODING:

```
1 usage
1  def is_perfect(n):
2      sum = 0
3      for i in range(1, n):
4          if n % i == 0:
5              sum += i
6      return sum == n
1 usage
7  def find_perfect_numbers(up_to):
8      perfect_numbers = []
9      for i in range(1, up_to + 1):
10         if is_perfect(i):
11             perfect_numbers.append(i)
12     return perfect_numbers
13  up_to = 10000
14  result = find_perfect_numbers(up_to)
15  print(f"Perfect numbers up to {up_to}: {result}")
```

OUTPUT:

```
C:\Users\vinot\PycharmProjects\pythonProject3\.venv\Script
Perfect numbers up to 10000: [6, 28, 496, 8128]

5
↓
Process finished with exit code 0
```

3. Write C program that demonstrates the usage of these notations by analyzing the time complexity of some example algorithms.

Coding:

```
# Example 1: O(n) time complexity
def example1(n):
    for i in range(1, n + 1):
        print("Hello World!!!")

# Example 2: O(log n) time complexity
def example2(n):
    i = 1
    while i <= n:
        print("Hello World!!!")
        i *= 2

# Example 3: O(n^2) time complexity
def example3(n, m):
    for i in range(n):
        for j in range(m):
            print("Hello World!!!")

# Example 4: O(log log n) time complexity
def example4(n):
    i = 2
    while i <= n:
        print("Hello World!!!")
        i **= 2

# Example 5: Exponential Time - O(2^n)
def fibonacci(n):
    if n <= 1:
        return n
    return fibonacci(n - 1) + fibonacci(n - 2)

# Example 6: Quadratic Time - O(n^2)
def bubble_sort(data):
    swapped = True
    while swapped:
        swapped = False
        for i in range(len(data) - 1):
            if data[i] > data[i + 1]:
                data[i], data[i + 1] = data[i + 1], data[i]

# Example 7: Big Theta Notation (Θ)
def merge_sort(arr):
    if len(arr) <= 1:
        return arr
    mid = len(arr) // 2
    left_half = arr[:mid]
    right_half = arr[mid:]
    return merge(merge_sort(left_half), merge_sort(right_half))

def merge(left, right):
    merged = []
    left_index = 0
    right_index = 0
    while left_index < len(left) and right_index < len(right):
        if left[left_index] <= right[right_index]:
```

Output:

```
C:\Users\vinot\PycharmProjects\pythonProject3\.ven
Hello World!!!
Hello World!!!
Hello World!!!
Hello World!!!
Hello World!!!
Hello World!!!
Hello World!!!
```

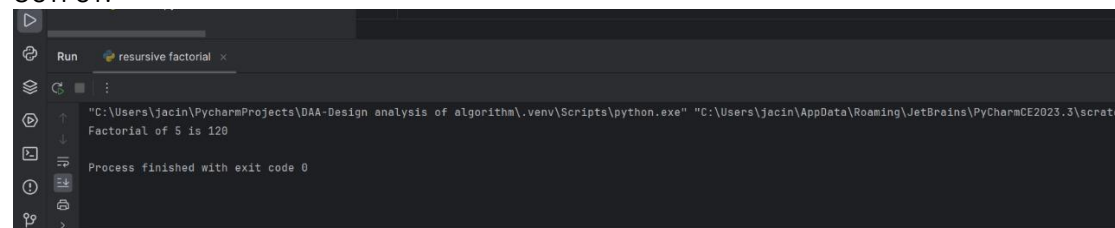
4. Write C programs that demonstrate the mathematical analysis of non-recursive and recursive algorithms.

NON RECURSIVE:

CODING:

```
1 def factorial_iterative(n):
2     result = 1
3     for i in range(1, n + 1):
4         result *= i
5     return result
6
7 # Example usage
8 n = 5
9 print("Factorial of", n, "is", factorial_iterative(n))
```

OUTPUT:



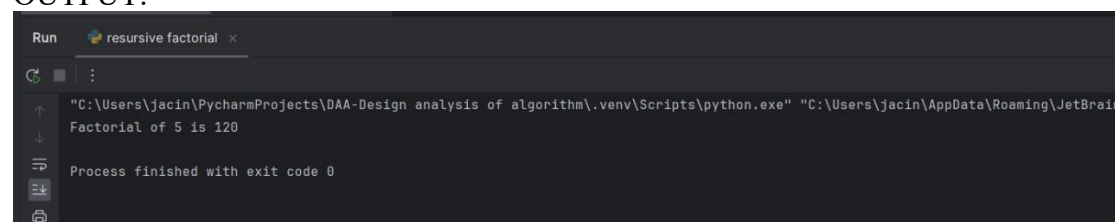
The screenshot shows a Python IDE with a 'Run' window. The output is 'Factorial of 5 is 120'. The process finished with exit code 0.

RECURSIVE:

CODING:

```
1 def factorial_recursive(n):
2     if n == 0:
3         return 1
4     else:
5         return n * factorial_recursive(n - 1)
6
7 # Example usage
8 n = 5
9 print("Factorial of", n, "is", factorial_recursive(n))
```

OUTPUT:



The screenshot shows a Python IDE with a 'Run' window. The output is 'Factorial of 5 is 120'. The process finished with exit code 0.

5. Write C programs for solving recurrence relations using the Master Theorem, Substitution Method, and Iteration Method will demonstrate how to calculate the time complexity of an example recurrence relation using the specified technique.

CODING:

```
#master theorem
from math import log2

def master_theorem(a, b, f_n):
    if a < b**f_n(1):
        return "O(" + str(f_n(1)) + ")"
    elif a == b**f_n(1):
        return "O(" + str(f_n(1)) + " log n)"
    else:
        return "O(" + str(a) + "^n)"

# Example usage:
a = 2
b = 2
f_n = lambda n: n # f(n) = n
print(master_theorem(a, b, f_n)) # Output: O(n log n)

#substitution method
def substitution_method(T_n, guess):
    n = 1
    while True:
        if T_n(n) == guess(n):
            n *= 2
        else:
            break
    return "O(" + str(guess(1)) + ")"

# Example usage:
T_n = lambda n: 2*T_n(n/2) + n # T(n) = 2T(n/2) + n
guess = lambda n: n*log2(n) # Guess: T(n) = n log n
print(substitution_method(T_n, guess)) # Output: O(n log n)

#iteration method
def iteration_method(T_n):
    n = 1
    iterations = 0
    while True:
        if T_n(n) == 1: # Base case
            break
        n *= 2
        iterations += 1
    return "O(" + str(2**iterations) + ")"

# Example usage:
T_n = lambda n: 2*T_n(n/2) + n # T(n) = 2T(n/2) + n
print(iteration_method(T_n)) # Output: O(n log n)
```

OUTPUT:

```
↑ C:\Users\saisr\AppData\Local\Microsoft
↓ "C:\Users\saisr\Downloads\assignments
0(1 log n)
```

6. Given two integer arrays nums1 and nums2, return an array of their Intersection. Each element in the result must be unique and you may return the result in any order.

CODING:

```
1 from typing import List
1 usage
2 class Solution:
3     3 usages
4     def intersection(self, nums1: List[int], nums2: List[int]) -> List[int]:
5         return list(set(nums1) & set(nums2))
6 # Test the code
7 solution = Solution()
8 nums1 = [1, 2, 2, 1]
9 nums2 = [2, 2]
10 print(solution.intersection(nums1, nums2)) # Output: [2]
11 nums1 = [4, 9, 5]
12 nums2 = [9, 4, 9, 8, 4]
13 print(solution.intersection(nums1, nums2)) # Output: [4, 9]
14 nums1 = [1, 2, 3, 4, 5]
15 nums2 = [6, 7, 8, 9, 10]
16 print(solution.intersection(nums1, nums2)) # Output: []
```

OUTPUT:

```
C:\Users\vinot\PycharmProjects\pythonProject
[2]
[9, 4]
[]

Process finished with exit code 0
```

7. Given two integer arrays nums1 and nums2, return an array of their intersection. Each element in the result must appear as many times as it shows in both arrays and you may return the result in any order.

CODING:

```

from collections import defaultdict

def intersect(nums1, nums2):
    # Count the occurrences of each element in nums1
    count = defaultdict(int)
    for num in nums1:
        count[num] += 1

    # Find the intersection and update the count
    result = []
    for num in nums2:
        if num in count and count[num] > 0:
            result.append(num)
            count[num] -= 1

    return result

# Example usage
nums1 = [1, 2, 2, 1]
nums2 = [2, 2]
print(intersect(nums1, nums2)) # Output: [2, 2]

```

Output:

The screenshot shows a Python IDE window titled 'Run' with a file named '7th.py'. The code being executed is the same as in the previous block. The output of the program is displayed in the console as '[2, 2]'. Below the output, a message states 'Process finished with exit code 0'.

8. Given an array of integers nums, sort the array in ascending order and return it. You must solve the problem without using any built-in functions in $O(n \log(n))$ time complexity and with the smallest space complexity possible.

CODING:


```

def merge_sort(nums):
    if len(nums) <= 1:
        return nums

    mid = len(nums) // 2
    left_half = nums[:mid]
    right_half = nums[mid:]

    left_half = merge_sort(left_half)
    right_half = merge_sort(right_half)

    return merge(left_half, right_half)

def merge(left, right):
    result = []
    left_index, right_index = 0, 0

    while left_index < len(left) and right_index < len(right):
        if left[left_index] < right[right_index]:
            result.append(left[left_index])
            left_index += 1
        else:
            result.append(right[right_index])
            right_index += 1

    result.extend(left[left_index:])
    result.extend(right[right_index:])

    return result

# Example usage
nums = [4, 2, 5, 1, 3]
sorted_nums = merge_sort(nums)
print(sorted_nums)

```

OUTPUT:

The screenshot shows a Python IDE window titled "Run" with a file named "master_throms.py". The output of the program is displayed in the console, showing the sorted list [1, 2, 3, 4, 5]. The console also shows the file path and the command used to run the program.

```

Run master_throms.py
"C:\Users\jacin\PycharmProjects\OAA-Design analysis of algorithm\venv\Scripts\python.exe" "C:\Users\jacin\AppData\Roaming\JetBrains\PyCharmCE2023.3\scratches\master_throms.py"
[1, 2, 3, 4, 5]
Process finished with exit code 0

```


9. Given an array of integers `nums`, half of the integers in `nums` are odd, and the other half are even.

CODING:

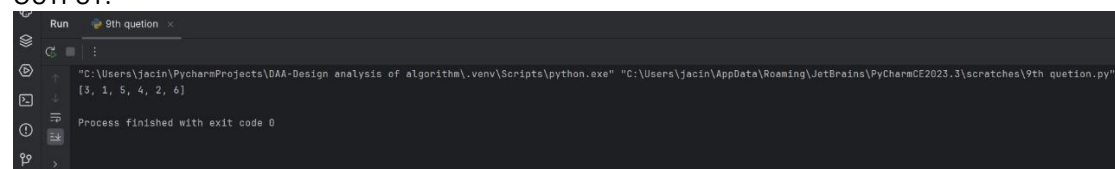
```
def partition_array(nums):
    left, right = 0, len(nums) - 1

    while left < right:
        if nums[left] % 2 == 0 and nums[right] % 2 == 1:
            nums[left], nums[right] = nums[right], nums[left]
            left += 1
            right -= 1
        elif nums[left] % 2 == 1:
            left += 1
        elif nums[right] % 2 == 0:
            right -= 1

    return nums

# Example usage
nums = [3, 1, 2, 4, 5, 6]
partitioned_nums = partition_array(nums)
print(partitioned_nums)
```

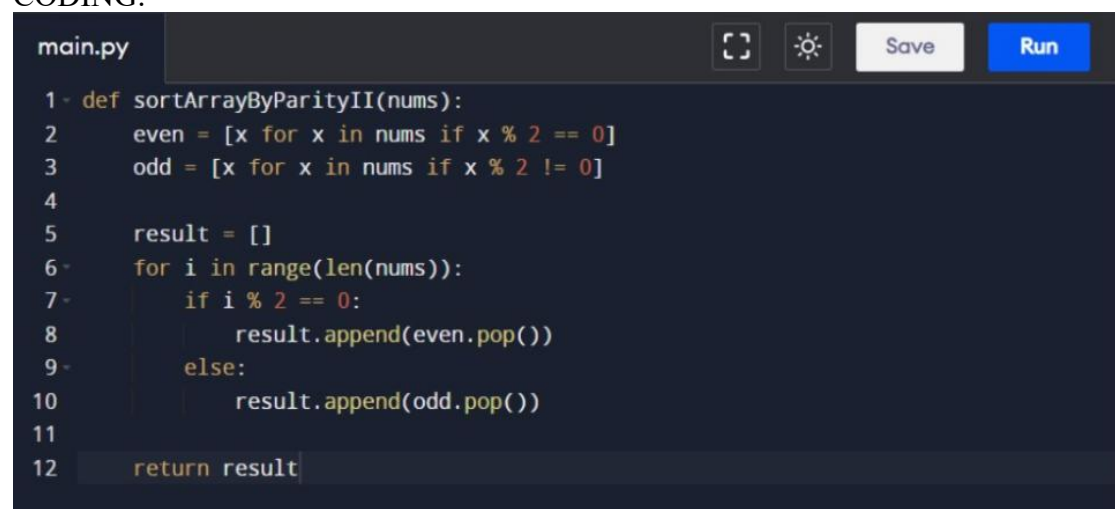
OUTPUT:



```
Run 9th question x
"C:\Users\jacin\PycharmProjects\DA-Design analysis of algorithms\.venv\Scripts\python.exe" "C:\Users\jacin\AppData\Roaming\JetBrains\PyCharmCE2023.3\scratches\9th_question.py"
[3, 1, 5, 4, 2, 6]
Process finished with exit code 0
```

10. Sort the array so that whenever `nums[i]` is odd, `i` is odd, and whenever `nums[i]` is even, `i` is even. Return any answer array that satisfies this condition.

CODING:



```
main.py
def sortArrayByParityII(nums):
    even = [x for x in nums if x % 2 == 0]
    odd = [x for x in nums if x % 2 != 0]

    result = []
    for i in range(len(nums)):
        if i % 2 == 0:
            result.append(even.pop())
        else:
            result.append(odd.pop())

    return result
```