**Reg. No: 21BCE1297 Name: Vidhi Shah Date: 22/03/22**

**PPS8**

**Q1**

**Aim:**

Simulate the MATH calculator for the following operators (+, \*, $) by using C functions.

‘+’ Operator does the following: Accept 2 integers as input and perform the summation of the given two numbers and return the sum as answer.

‘\*’ Operator: Accepts 2 integers as input and multiply the given two numbers and return the product as answer.

‘$’ Operator: Accept an integer as input and return the reverse of the given number.

**Procedure:**

**Input:**

Operator

Operands

**Output:**

Addition, multiplication or reverse of the number

**Algorithm:**

Step 1: Declare ‘add’, ‘multiplication’ and ‘reverse’ function with return type ‘int’ and argument ‘void’

**Main Function**

Step 1: Read operator, op

Step 2: Use switch case for operator

Case 1 (‘+’): Call function ‘add’ and print its return value

Case 2 (‘\*’): Call function ‘multiplication’ and print its return value

Case 3 (‘$’): Call function ‘reverse’ and print its return value

Default: Print error message for invalid input

**Add Function**

Step 1: Declare integer variables ‘a’, ‘b’ and ‘sum’

Step 2: Read ‘a’ and ‘b’

Step 3: sum = a + b

Step 4: return sum

**Multiply Function**

Step 1: Declare integer variables ‘a’, ‘b’ and ‘prod’

Step 2: Read ‘a’ and ‘b’

Step 3: prod = a \* b

Step 4: return prod

**Reverse Function**

Step 1: Declare integer variable ‘a’ and ‘rev’

Step 2: Read ‘a’

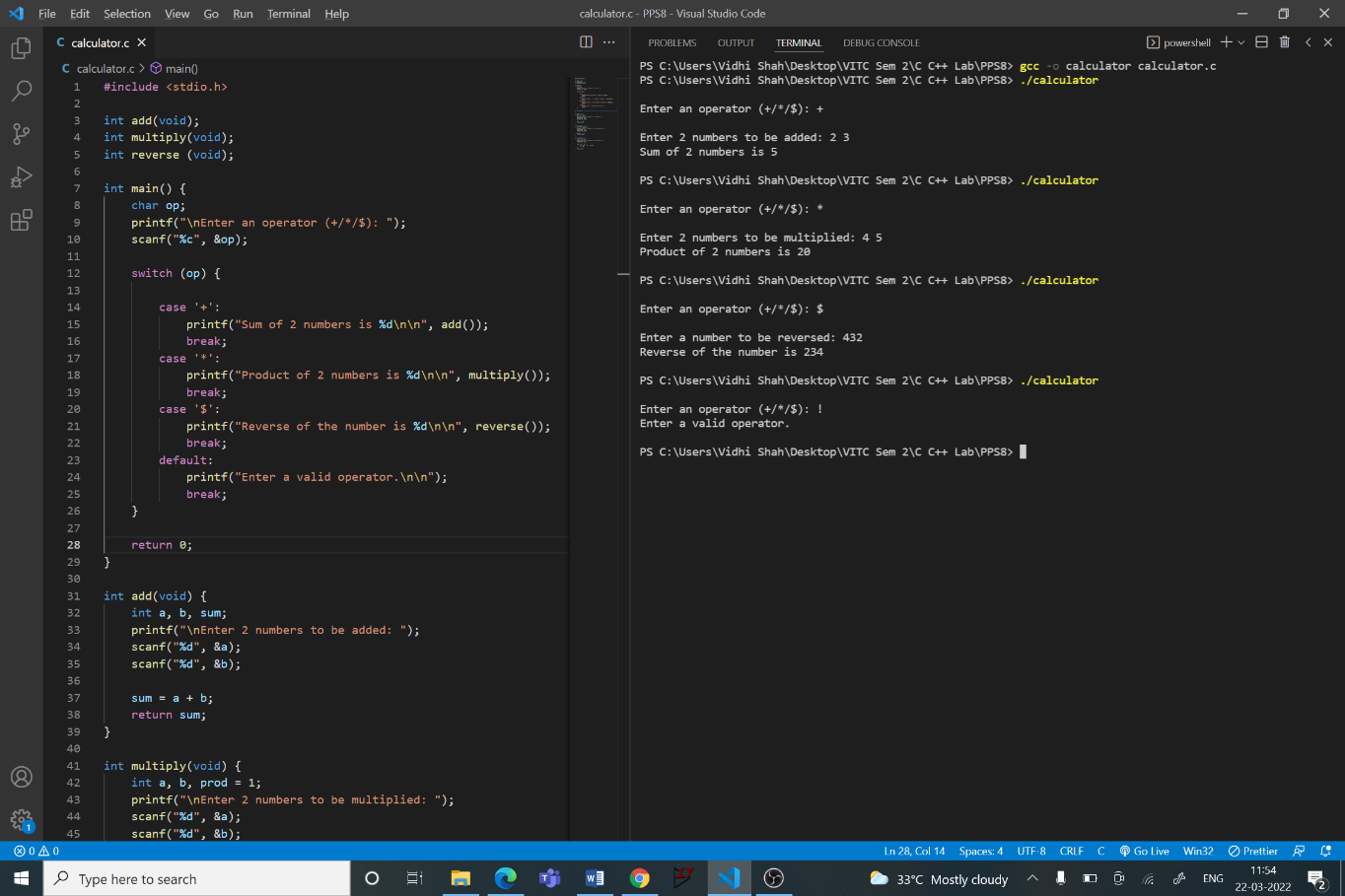
Step 3: While n > 0

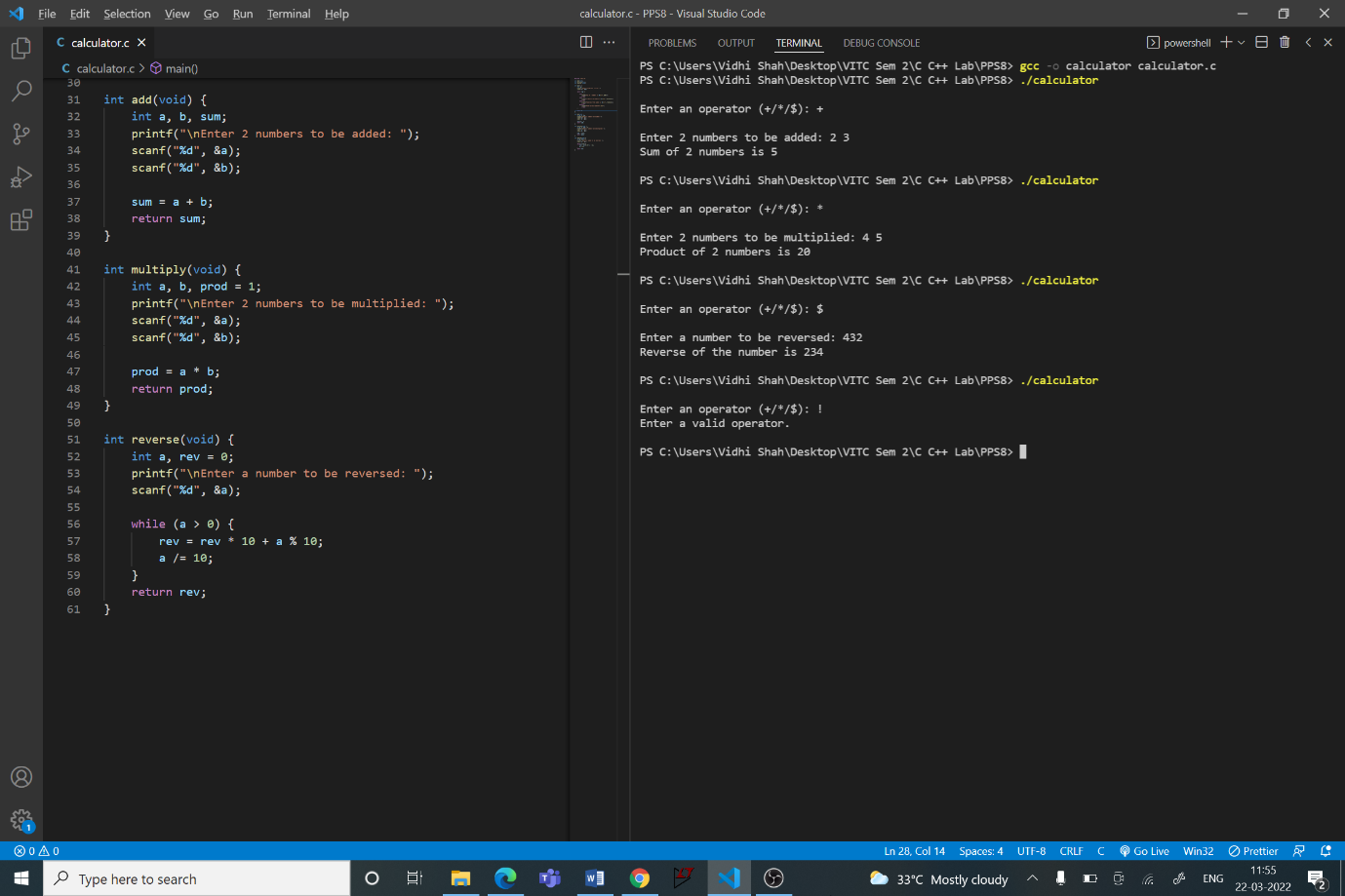
rev = (rev \* 10) + (a % 10)

a = a / 10

Step 4: return rev

**Code:**

****

****

**Q2**

**Aim:**

Given an array, write a function to reverse the first and second half of an array keeping its center element unchanged if the array elements are odd.

**Procedure:**

**Input:**

Number of elements in the array, ‘n’

‘Array of length ’n’

**Output:**

Array with 2 halves reversed separately

**Algorithm:**

Step 1: Declare ‘reverse’ function with return type ‘void’ and argument ‘void’

**Main Function**

Step 1: Call function ‘reverse’

**Reverse Function**

Step 1: Initialise integer variable ‘n’ and ‘i’

Step 2: Read ‘n’

Step 3: Initialise 2 integer arrays of size n, ‘arr’ and ‘rev’. Let mid = n/2

Step 4: Read elements into the array

Step 5: For ‘i’ from 0 to mid (excluding mid)

Step A: rev[mid - i - 1] = arr[i]

Step 6: If n is odd

Step A: rev[mid] = arr[mid]

Step B: For ‘i’ from 0 to mid (excluding mid)

Step B1: rev[n - i - 1] = arr[mid + i + 1]

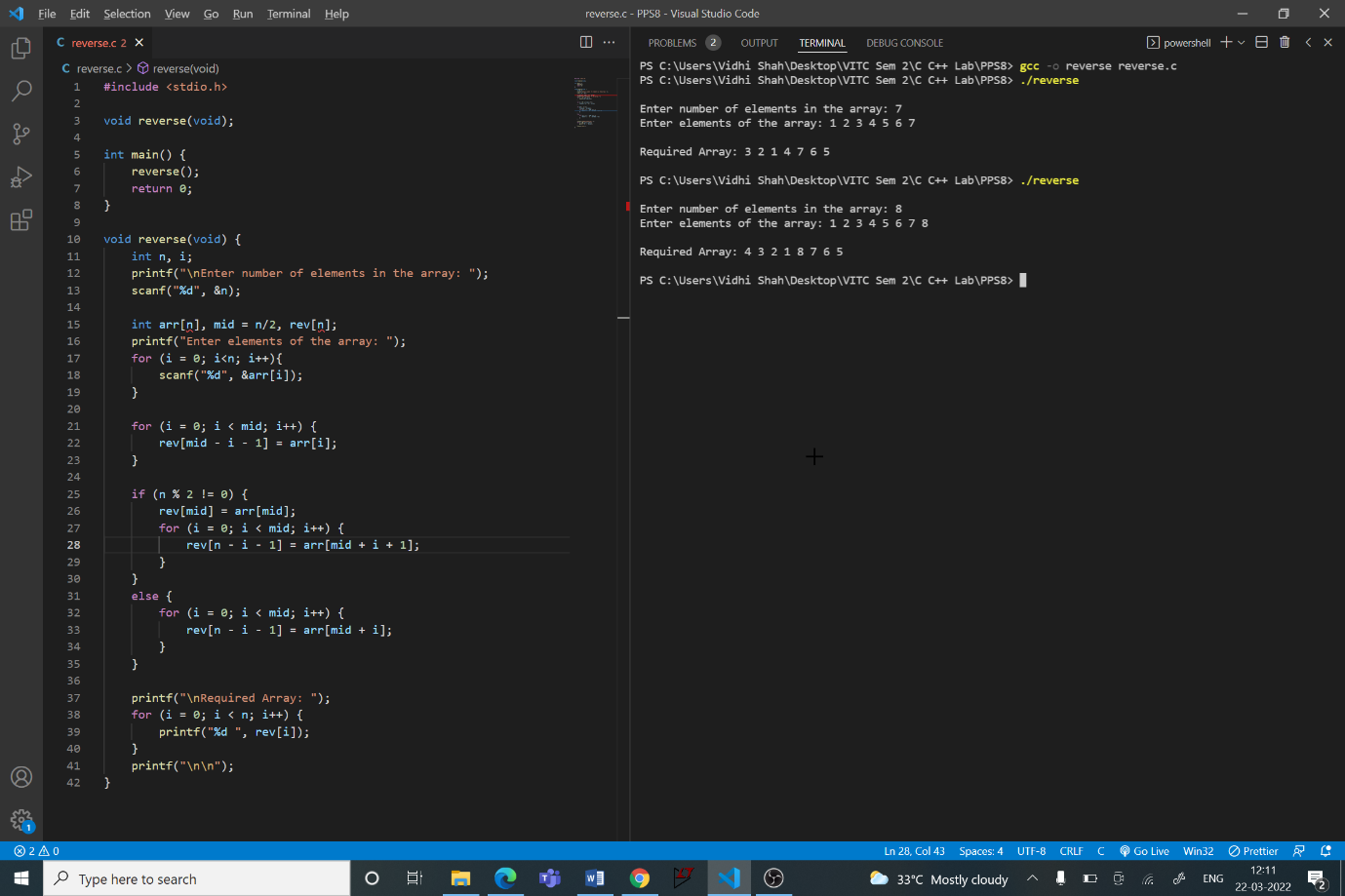
Step 6: If n is even

Step A: For ‘i’ from 0 to mid (excluding mid)

Step A1: rev[n - i - 1] = arr[mid + i]

Step 7: Display the rev array

**Code:**

****

**Q3**

**Aim:**

Given an array, write a function to insert a number ‘x’ at index 0.

**Procedure:**

**Input:**

Number of elements in the array, ‘n’

Array of length ‘n’

A number, ‘x’

**Output:**

Array with ‘x’ at 0th index

**Algorithm:**

Step 1: Declare ‘insert0’ function with return type ‘void’ and argument ‘void’

**Main Function**

Step 1: Call function ‘insert0’

**Insert0 Function**

Step 1: Initialise integer variable ‘n’, ‘i’ and ‘x’

Step 2: Read ‘n’

Step 3: Initialise an integer array of size n + 1

Step 4: Read elements into the array

Step 5: Read ‘x’

Step 6: Increment n by 1

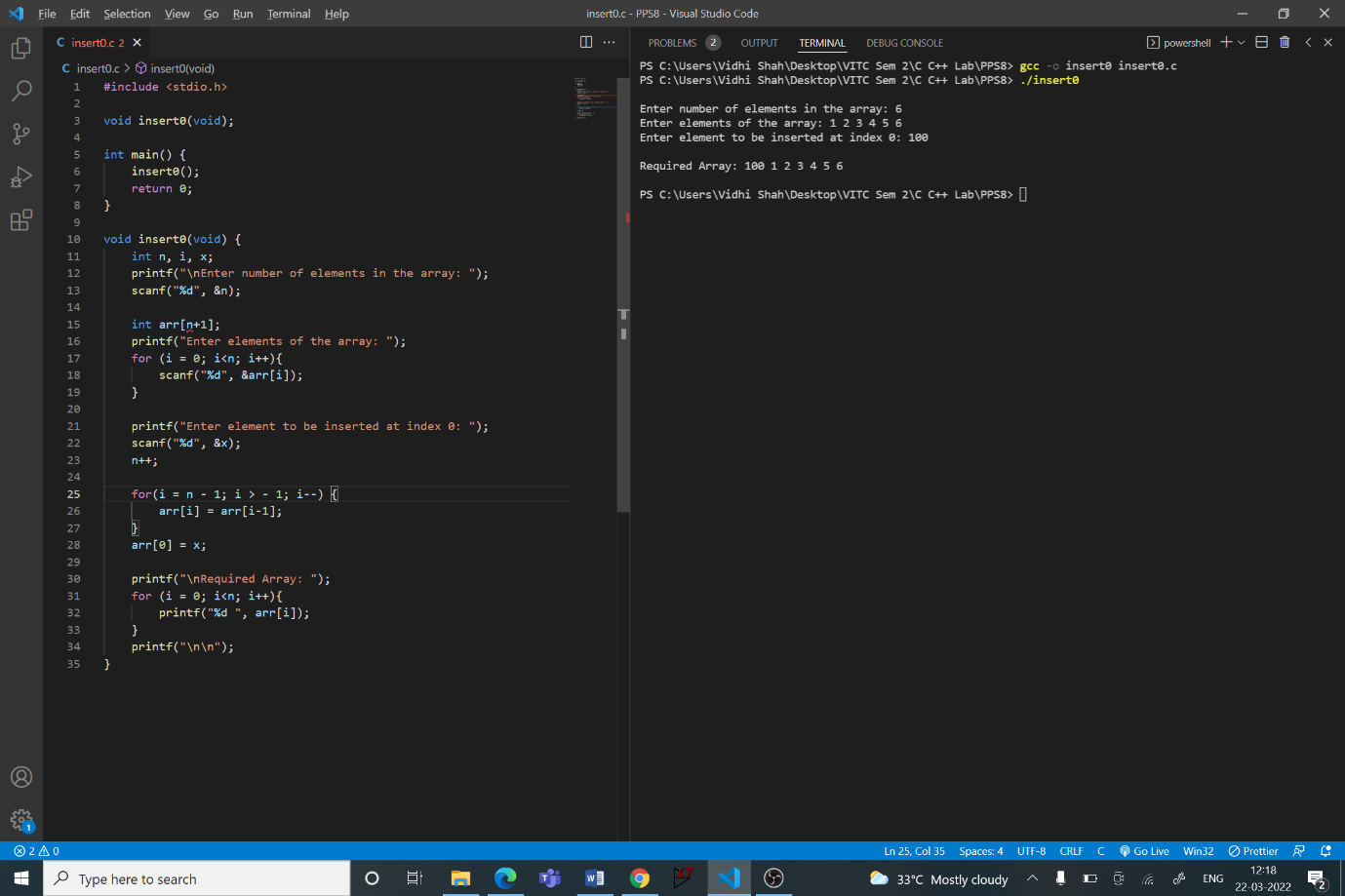
Step 7: For ‘i’ from n - 1 to -1 (excluding -1)

Step A: arr[i] = arr[i-1]

Step 8: arr[0] = x

Step 9: Display the array

**Code:**

****

**Q4**

**Aim:**

Given an array and a new position ‘p’, write a function to insert a number ‘x’ at position ‘p’

**Procedure:**

**Input:**

Number of elements in the array, ‘n’

Array of length ‘n’

A position, ‘p’

A number, ‘x’

**Output:**

Array with ‘x’ at pth index

**Algorithm:**

Step 1: Declare ‘insert’ function with return type ‘void’ and argument ‘void’

**Main Function**

Step 1: Call function ‘insert’

**Insert Function**

Step 1: Initialise integer variable ‘n’, ‘i’, ‘p’ and ‘x’

Step 2: Read ‘n’

Step 3: Initialise an integer array of size n + 1

Step 4: Read elements into the array

Step 5: Read ‘p’ and ‘x’

Step 6: Increment n by 1

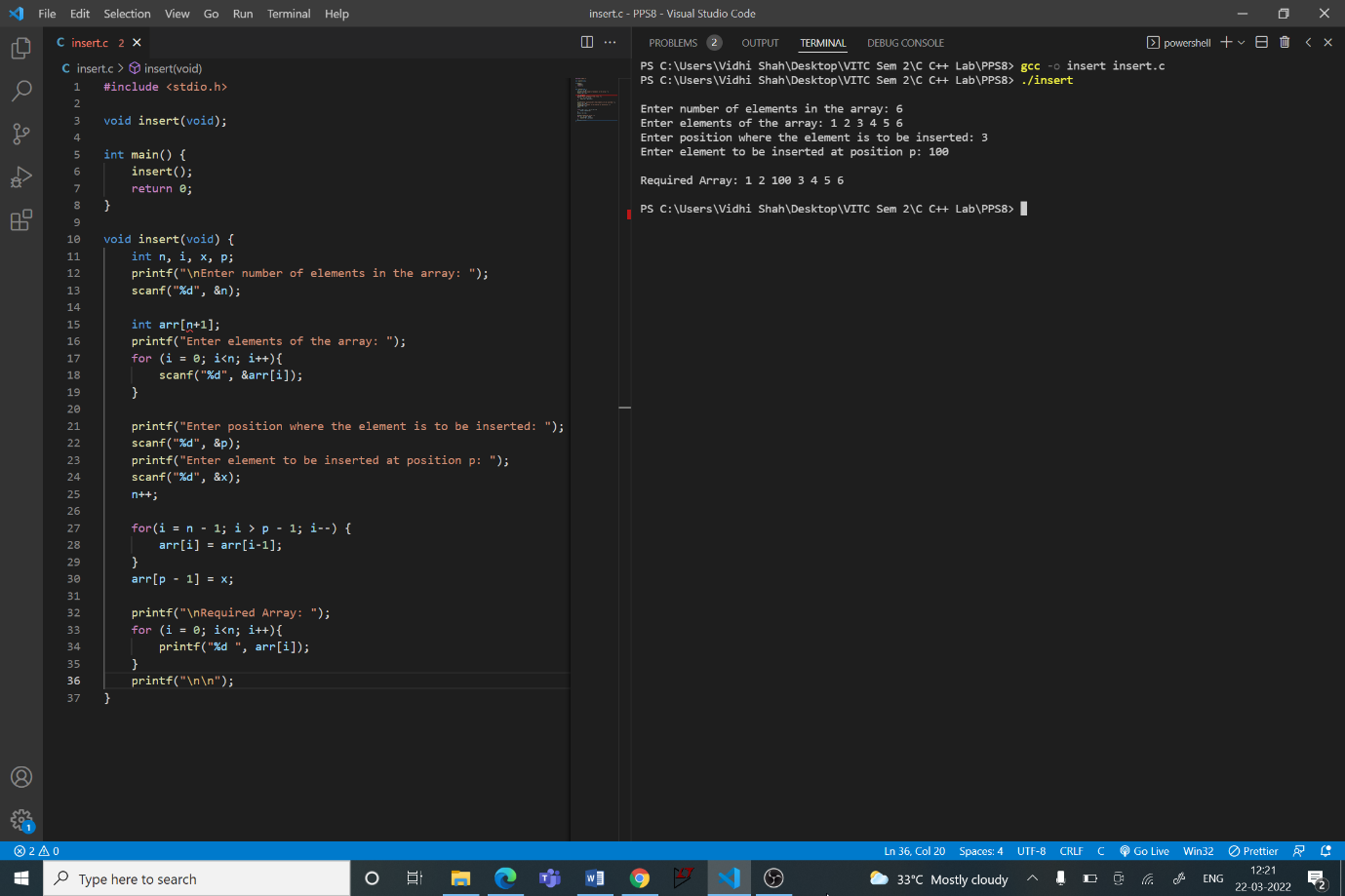
Step 7: For ‘i’ from n - 1 to p - 1 (excluding p - 1)

Step A: arr[i] = arr[i-1]

Step 8: arr[p - 1] = x

Step 9: Display the array

**Code:**

****