Ques1.1 Write a program to find divisor or factorial of a given number.

Step 1: Start
Step 2: Declare variables num1, fact, i.
Step 3: Read value of num1
Step 4: Declare fact to 1 and i to 1
Step 5: repeat steps 6 to 8 until i <= num1
Step 6: is num1%i == 0? If yes then goto step 7 else go to step 8.
Step 7: Print i (Divisor of num1)
Step 8: increment i
Step 9: declare i to 1
Step 10: repeat steps 11 to 12 until i <= num1
Step 11: update fact as fact = fact*i
Step 12: increment i
Step 13: Print fact (factorial of num1)
Step 14: End

Ques1.2 Write a program to find sum of geometric series.

Step 1: Start
Step 2: Declare variables a, r, n, i, sum.
Step 3: Read value of a (First term), r (common ratio), n (number of terms).
Step 4: Declare sum to 0 and i to 0
Step 5: Repeat steps 6 to 8 until i<n
Step 6: update sum as sum = sum + a
Step 7: update a as a = a*r
Step 8: increment i
Step 9: Print sum
Step 10: End

Ques1.3 Write recursive program to print the first m Fibonacci number.

Step 1: Start Main
Step 2: Declare variables n, i
Step 3: Read value of n (Number of fibonacci series)
Step 4: Repeat steps 5 to 6 until i<n
Step 5: print fibonacci(i)
Step 6: increment i
Step 7: End Main

Step 8: start fibonacci(integer n)
Step 9: declare variable i
Step 10: is n==0? If yes then goto step 11 else goto step 13
Step 11: declare i to 0
Step 12: goto step 17
Step 13: is n==1? If yes then goto step 14 else goto step 16
Step 14: update i=1
Step 15: goto step 17
Step 16: update i = fibonacci(n-1) + fibonacci(n-2)
Step 17: return i
Step 18: end fibonacci

Ques1.4 Write a menu driven program for matrices to do the following operations depending on whether the operations require one or two matrices

- **Addition of 2 Matrices**
- **Subtraction of 2 Matrices**
- **Finding upper and lower triangular Matrices**
- **Transpose of a Matrix**
- **Product of 2 Matrices**

Step 1: Start
Step 2: Declare variables option and opt
Step 3: Read value of option to choose between Adding, Subtracting, finding upper or lower triangular matrix, finding transpose or multiplying matrices. (1,2,3,4,5,6,…etc)
Step 4: Is option=1? If yes call sum()
Step 5: Is option=2? If yes call minus()
Step 6: Is option=3? If yes call upper()
Step 7: Is option=4? If yes call lower()
Step 8: Is option=5? If yes call transpose()
Step 9: Is option=6? If yes call multiply()
Step 10: Read value of opt to choose if to run the program again
Step 11: Is opt=1? If yes call main()
Step 12: Is opt=0? If End Main
Step 13: End Main

Step 14: Start sum
Step 15: declare variables k, i, j, row, column
Step 16: Read values of row and column
Step 17: Declare 2 two dimensional arrays arr1 and arr2 with size row x column
Step 18: Read value of Arrays arr1 and arr2
Step 19: declare two dimensional array SUM with size row x column
Step 18: declare i as 0
Step 21: repeat Step 22 to 26 until i<row
Step 22: declare j as 0
Step 23: repeat Step 24 to 25 until j<column
Step 24: compute SUM[i][j] = arr1[i][j] + arr2[i][j];
Step 25: increment j
Step 26: increment i
Step 27: Print SUM array
Step 28: end sum

Step 29: Start minus
Step 30: declare variables k, i, j, row, column
Step 31: Read values of row and column
Step 32: Declare 2 two dimensional arrays arr1 and arr2 with size row x column
Step 33: Read value of Arrays arr1 and arr2
Step 34: declare two dimensional array MINUS with size row x column
Step 35: declare i as 0
Step 36: repeat Step 37 to 41 until i<row
Step 37: declare j as 0
Step 38: repeat Step 39 to 40 until j<column
Step 39: compute MINUS[i][j] = arr1[i][j] - arr2[i][j];
Step 40: increment j
Step 41: increment i

Step 42: Print MINUS array
Step 43: end minus

Step 44: Start upper
Step 45: declare variables k, i, j, row, column
Step 46: Read values of row and column
Step 47: Declare two dimensional array arr with size row x column
Step 48: Read value of Array arr
Step 49: declare two dimensional array UPP with size row x column
Step 50: declare i as 0
Step 51: repeat Step 52 to 56 until i<row
Step 52: declare j as 0
Step 53: repeat Step 54 to 55 until j<column
Step 54: Is i>j? If yes update value at UPP[i][j] as 0 else update value at UPP[i][j] as arr[i][j]
Step 55: increment j
Step 56: increment i
Step 57: Print UPP array (upper triangular matrix)
Step 58: End upper

Step 59: Start lower
Step 60: declare variables k, i, j, row, column
Step 61: Read values of row and column
Step 62: Declare two dimensional array arr with size row x column
Step 63: Read value of Array arr
Step 64: declare two dimensional array LOW with size row x column
Step 65: declare i as 0
Step 66: repeat Step 67 to 71 until i<row
Step 67: declare j as 0
Step 68: repeat Step 69 to 70 until j<column
Step 69: Is i<j? If yes update value at LOW[i][j] as 0 else update value at LOW[i][j] as arr[i][j]
Step 70: increment j
Step 71: increment i
Step 72: Print LOW array (lower triangular matrix)
Step 73: End lower

Step 74: Start transpose
Step 75: declare variables k, i, j, row, column
Step 76: Read values of row and column
Step 77: Declare two dimensional array arr with size row x column
Step 78: Read value of Array arr
Step 79: declare two dimensional array TRANS with size row x column
Step 80: declare i as 0
Step 81: repeat Step 82 to 86 until i<row
Step 82: declare j as 0
Step 83: repeat Step 84 to 85 until j<column
Step 84: update value of TRANS[j][i] as arr[i][j]
Step 85: increment j
Step 86: increment i
Step 87: Print TRANS array (Transpose of a matrix)
Step 88: End transpose

Step 89: Start multiply
Step 90: declare variables k, i, j, row1, column1, row2, column2
Step 91: Read values of row1, column1, row2 and column2
Step 92: Is column1 == row2? If yes goto step 93 else print 'Multiplication not possible' and end multiply
Step 93: Declare 2 two dimensional arrays arr1 and arr2 with size row1 x column1 and row2 x column2 each
Step 94: Read value of Arrays arr1 and arr2
Step 95: declare two dimensional array MULTI with size row1 x column2
Step 96: declare i as 0
Step 97: repeat Step 98 to 106 until i<row1
Step 98: declare j as 0
Step 99: repeat Step 100 to 105 until j<column2
Step 100: update value at MULTI[i][j] as 0
Step 101: declare k as 0
Step 102: repeat Step 103 to 104 until j<column1
Step 103: update value of MULTI[i][j] as MULTI[i][j] + arr1[i][k] * arr2[k][j]
Step 104: increment k
Step 105: increment j
Step 106: increment i
Step 107: Print MULTI array (Product of two matrices)
Step 108: End multiply

Ques1.5 Write a program to perform the following operators on Strings without using String functions

- **To find the Length of String.**
- **To concatenate the string.**
- **To find Reverse of a string.**
- **To copy one string to another string**

Step 1: Start
Step 2: Declare variables option, i, j, len.
Step 3: Declare string str1 and str2 of size 500 characters
Step 4: Read value of str1.
Step 5: Declare len to 0 and j to 0
Step 6: Increment len untill value of str1 at index 'len' is not null
Step 7: Read value of option to choose between finding length, concatenating, reversing or copying string. (1,2,3,4,…etc)
Step 8: is option=1? If yes print len and goto step 35 else goto step 9
Step 9: Is option=2? If yes goto step 10 else goto step 18
Step 10: read value of str2
Step 11: declare i to 1
Step 12: Repeat steps 13 to 14 until value of str2 at index i is not null
Step 13: update value of str1 at index 'len+i' as value of str2 at index i
Step 14: Increment i
Step 15: update value of str1 at index 'len+i' as null
Step 16: Print str1 (concatenated string)
Step 17: Goto step 35
Step 18: Is option=3? If yes goto step 19 else goto step 26
Step 19: declare i as len-1
Step 20: Repeat step 21 to 24 until i>=0

Step 21: update value of str2 at index j as value of str1 at index i
Step 22: decrement i and increment j
Step 23: update value of str2 at index j as null
Step 24: print str2 (reversed string)
Step 25: Goto step 35
Step 26: Is option=4? If yes goto step 27 else goto step 34
Step 27: declare i as 0
Step 28: Repeat step 29 to 30 until value of str1 at index i is not null
Step 29: update value of str2 at index i as value of str1 at index i
Step 30: increment i
Step 31: update value of str2 at index i as null
Step 32: print str2 (copied string)
Step 33: Goto step 30
Step 34: Print "Invalid Input"
Step 35: End

Ques2.1 Write an algorithm that reads the two numbers and print the value of the largest number. Also draw the flowchart using Flowgorithm.

Step 1: Start
Step 2: Declare variables num1, num2.
Step 3: Read values of num1 and num2.
Step 4: Is num1>num2? If yes print num1 else print num2.
Step 5: End

Ques2.2 Write an algorithm and draw a flowchart to find the sum of two numbers

Step 1: Start
Step 2: Declare variables num1, num2 and sum.
Step 3: Read values of num1 and num2.
Step 4: Compute num1 + num2 and assign its value to sum.
Step 5: print sum
Step 6: End

Ques2.3 Write a C program to print Hello world. Also draw the flowchart using Flowgorithm.

Step 1: Start
Step 2: Output "Hello, World!"
Step 3: End

Ques2.4 Write a program and draw a flowchart to check whether a number is even or odd.

Step 1: Start
Step 2: Declare Variable num1.

Step 3: Read value of num1.
Step 4: Is num1%2==0? If yes then print “Even” else print ”odd”
Step 5: End

Ques2.5 10% discount is given, when a customer buys more than 100 items. Item cost will be entered by the user. Write an algorithm to calculate the final cost that has to be paid. Also draw the flowchart using flowgorithm.
Step 1: Start
Step 2: Declare Variable quantity, mrp, cost
Step 3: Read value of mrp, quantity
Step 4: update cost as mrp * quantity
Step 5: Is quantity>100? If yes then goto step 6 else goto step 7
Step 6: update cost as cost – (cost/10)
Step 7: Print cost
Step 8: End

Ques2.6 Write a C Program to print pyramid of *.
Step 1: Start
Step 2: Declare Variable i, space, rows, k
Step 3: Read value of rows
Step 4: Declare i to 0 and k to 0
Step 5: Repeat steps 6 to 14 until i<=rows
Step 6: declare space to 1
Step 7: Repeat step 8 to 9 until space<=rows-i
Step 8: print “ “ on same line
Step 9: Increment space
Step 10: Repeat steps 11 to 12 until k!=2*i-1
Step 11: Print “* “ on same line
Step 12: Increment k
Step 13: Print new line
Step 14: Increment i
Step 15: End

Ques2.7 Write a C Program to print inverted pyramid of *.
Step 1: Start
Step 2: Declare Variable i, space, rows, j
Step 3: Read value of rows
Step 4: Declare i to rows
Step 5: Repeat steps 6 to 19 until i>=1
Step 6: Declare space to 0
Step 7: Repeat step 8 to 9 until space<rows-i
Step 8: print “ “ on same line
Step 9: Increment space
Step 10: declare j to i
Step 11: Repeat steps 12 to 13 until j<=2*i-1
Step 12: Print “* “ on same line

Step 13: Increment j
Step 14: Declare j to 0
Step 15: Repeat steps 16 to 17 until j<i-1
Step 16: Print "* " on same line
Step 17: Increment j
Step 18: Print new line
Step 19: Increment i
Step 20: End

Ques2.8a Write a C program to understand the concept of continue statement and calculate sum of numbers. Also draw the flowchart of the given program. If a user enters a negative number, it is not added into result.

Step 1: Start
Step 2: Declare Variable numoitem, num, i, sum
Step 3: Read value of numoitem (number of items)
Step 4: Declare i to 1 and sum to 0
Step 5: Repeat steps 6 to 10 until i<=numoitem
Step 6: Read value of num
Step 7: is num<0? If yes goto step 8 else goto step 9
Step 8: continue – Goto next incrementation
Step 9: update sum as sum + num
Step 10: Increment i
Step 11: Print sum
Step 12: End

Ques2.8b Write a C program to convert decimal number to binary number.

Step 1: Start
Step 2: Declare Variables i, j, k, dec, len
Step 3: Declare array bin of integer data type of size 64 – bin[64]
Step 4: Read value of dec (decimal number)
Step 5: Declare i to 1 and k to 0
Step 6: Repeat steps 7 to 9 until i!=0
Step 7: update value of bin at index k as i%2
Step 8: increment k
Step 9: update i as i/2
Step 10: declare j to k-1
Step 11: Repeat steps 12 to 13 until j>=0
Step 12: Print element of bin at index j
Step 13: decrement j
Step 14: End

Ques2.9 Write a C program to find sum of numbers and average of all numbers using arrays(i.e C program to read N integers into an array A and

a) Find the sum of all numbers.
b) Find the average of all numbers and display the result with suitable headings.
Step 1: Start
Step 2: Declare Variables i, item, sum, avg
Step 3: Read value of item (number of elements in array)
Step 4: Declare array arr of integer data type of size equal to value of item variable – bin[item]
Step 5: Declare i to 0
Step 6: Repeat steps 6 to 8 until i<item
Step 7: read and store value at arr at index i
Step 8: increment i
Step 9: declare i to 0
Step 10: Repeat steps 11 to 12 until i<item
Step 11: update sum as sum + element of arr at index i
Step 12: increment i
Step 13: compute sum/item and store the value to avg
Step 14: Print sum and avg
Step 15: End

Ques2.10 Write a C program to find sum of Natural numbers using recursion.
Step 1: Start Main
Step 2: Declare num
Step 3: Read value of num (Number of natural numbers)
Step 4: print add(num)
Step 5: End Main

Step 6: start add(integer n)
Step 7: declare variable i
Step 8: is n==0? If yes then goto step 9 else goto step 11
Step 9: update i = n
Step 10: goto step 12
Step 11: update i as n + add(n-1)
Step 12: return i
Step 13: end add

Ques2.11 Write a C program to understand the concept of call by value and call by reference.
Step 1: Start Main
Step 2: Declare a, b
Step 3: Declare a to 10 and b to 20
Step 4: print a and b (before calling any function)
Step 5: call swapByVal(a,b)
Step 6: print a and b (after calling swap by values)
Step 7: call swapByRef(memory location of a, memory location of b)
Step 8: print a and b (after calling call by reference)

Step 9: End Main

Step 6: start swapByVal(integer a, integer b)
Step 7: swap local variable a and b
Step 8: print a and b (in call by value function)
Step 9: end swapByVal

Step 10: start swapByRef(integer pointer a, integer pointer b)
Step 11: swap values at pointers a and b
Step 12: print a and b (in call by reference function)
Step 13: end swapByRef