**Roll No…………….. Total No. of Pages:……**

**FUNDAMENTALS OF C PROGRAMMING**

**Time allowed: 90 Minutes Max. Marks: 40**

**General Instructions:**

* **Follow the instructions given in each section.**
* **Make sure that you attempt the questions in order.**

**SECTION-A (10\*1 mark=10 marks)**

***(All questions are compulsory)***

Q1 What is the difference between a global variable and a local variable in C language?

A) Global variables can be accessed from any function, local variables can only be accessed from the function in which they are declared \*(Correct option)

B) Local variables can be accessed from any function, global variables can only be accessed from the function in which they are declared

C) Both global and local variables are the same

D) None of the above

Q2 What is the syntax for a do-while loop in C language?

A) while(condition){}

B) do{} while(condition) \*(Correct option)

C) for(;;){}

D) None of the above

Q3 What is the purpose of a goto statement in C language?

A) To transfer control to another part of the program \*(Correct option)

B) To end a program

C) To repeat a set of statements

D) None of the above

Q4 What is the purpose of the switch statement in C language?

A) To execute a set of statements based on the value of a single expression \*(Correct option)

B) To execute a set of statements based on the value of multiple expressions

C) To repeat a set of statements

D) None of the above

Q5 What is the data type used to store a single character in C language?

A) int

B) float

C) char \*(Correct option)

D) double

Q6 What is the difference between an if statement and a switch statement in C language?

A) If statements can handle multiple conditions, switch statements can only handle one condition \*(Correct option)

B) Switch statements can handle multiple conditions, if statements can only handle one condition

C) Both if statements and switch statements can handle multiple conditions

D) None of the above

Q7 What is the purpose of a pointer in C language?

A) To store the address of a variable \*(Correct option)

B) To store the value of a variable

C) To manipulate strings

D) None of the above

Q8 What is the purpose of the strlen() function in C language?

A) To determine the length of a string \*(Correct option)

B) To compare two strings

C) To copy a string to another string

D) None of the above

Q9 What is the purpose of the puts() function in C language?

A) To write a string to the screen \*(Correct option)

B) To read a string from the screen

C) To compare two strings

D) None of the above

Q10 What is the purpose of a union in C language?

A) To store different data types in the same memory location \*(Correct option)

B) To store the same data type in different memory locations

C) To store different data types in different memory locations

D) None of the above

**SECTION-B (5\*2 mark=10 marks)**

***(All questions are compulsory)***

11. Consider the following code in C:

int i = 1;

do {

printf("%d ", i);

i = i + 2; } while (i <= 5);

What is the output of the code?

a) 1 3 5 \*(Correct option)

b) 2 4 6

c) 1 2 3 4 5

d) 5 4 3 2 1

12. Can a function call itself in C programming?

a) Yes, this is known as recursion \*(Correct option)

b) No

c) Only if the function is defined as static

d) Only if the function is defined as global

13.What will be the output of the following code?

int x = 50;

int \*ptr1 = &x;

int \*ptr2 = &x;

int y = ++\*ptr2 ;

printf("%d %d", ++\*ptr1, y);

a) 52 51 \*(Correct option)

b) 50 50

c) 51 52

d) 51 50

14.What will be the output of the following code?

int arr[2][3] = {{1, 2, 3}, {4, 5, 6}};

printf("%d", \*(\*(arr+1)+2));

a) 1

b) 2

c) 3

d) 6 \*(Correct option)

15.What will be the output of the following code?

char name[10] = "Hello";

printf("%d", strlen(name));

a) 5 \*(Correct option)

b) 10

c) 6

d) 4

**SECTION-C(Coding Question) (2x5 marks=5 marks)**

Q16. Chaitanya has a N\*N matrix. He wants to print the jth anti-diagonal. jth anti-diagonal means a diagonal which starts from the (0,j)th cell and goes in the left-diagonal direction.

Give a N\*N square matrix, return an array of its anti-diagonals. Look at the example for more details.

**Input:**

The first line contains two integers N and j.

Second line contain a matrix of size N\*N.

**Constraints:**

1 <= N<= 10000 <= A[i][j] <= 10^5

0<=j<N

**Output:**

Print all elements of jth anti diagonal space separated in a single line.

Sample test Cases

|  |  |  |
| --- | --- | --- |
|  | Input | Output |
| STC1 | 5 2  1 2 3 4 5  2 3 4 5 1  8 7 0 9 8  6 7 8 5 6  6 17 2 12 2 | 3 3 8 |
| STC2 | 3 0  0 9 8  8 5 6  2 12 2 | 0 |

**Solution 16:**

#include<stdio.h>

int max(int num1, int num2)

{

return (num1 > num2 ) ? num1 : num2;

}

void solve(int n,int j, int a[][n])

{

int i=0;

while(j>=0 && i<n)

{

printf("%d ",a[i][j]);

i++;

j--;

}

}

int main()

{

int n,j;

scanf("%d%d",&n,&j);

int a[n][n];

for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

scanf("%d",&a[i][j]);

}

}

solve(n,j,a);

return 0;

}

**Test Cases**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Test Case 1 | Test Case 2 | Test Case 3 |
| Input | 2 1  0 1  1 9 | 3 2  1 2 3  5 4 3  2 3 5 | 1 0  1 |
| Output | 1 1 | 3 4 2 | 1 |

Q17. Write a program that takes a positive integer n from the user and checks whether that number can be expressed as the sum of two prime numbers.

If the number can be expressed as the sum of two prime numbers, the output shows the combination of the prime numbers otherwise display a message as “number cannot be expressed as the sum of two prime numbers.”

**Input:**

n is the positive number.

**Constraints:**

2 <= n <= 10^3

**Output:**

combination of the prime numbers. Example for n is 4 output is 4 = 2 + 2

**Solution**

#include <stdio.h>

int checkPrime(int n);

int main() {

int n, i, flag = 0;

printf("Enter a positive integer: ");

scanf("%d", &n);

for (i = 2; i <= n / 2; ++i) {

// condition for i to be a prime number

if (checkPrime(i) == 1) {

// condition for n-i to be a prime number

if (checkPrime(n - i) == 1) {

printf("%d = %d + %d\n", n, i, n - i);

flag = 1;

}

}

}

if (flag == 0)

printf("%d cannot be expressed as the sum of two prime numbers.", n);

return 0;

}

// function to check prime number

int checkPrime(int n) {

int i, isPrime = 1;

// 0 and 1 are not prime numbers

if (n == 0 || n == 1) {

isPrime = 0;

}

else {

for(i = 2; i <= n/2; ++i) {

if(n % i == 0) {

isPrime = 0;

break;

}

}

}

return isPrime;

}

**Test Cases**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Test Case 1 | Test Case 2 | Test Case 3 |
| Input | 2 | 34 | 4 |
| Output | number cannot be expressed as the sum of two prime numbers. | 34 = 3 + 31  34 = 5 + 29  34 = 11 + 23  34 = 17 + 17 | 4 = 2 + 2 |

**SECTION-D (Coding Question)(1x10 mark=10 mark)**

Q18 **Problem Statement: Write a function in C to allocate dynamic memory for a three-dimensional array of integers, initialize its elements and then sort its elements in ascending order using a doubly pointer.**

**Sample Input**:

x = 2, y = 3, z = 4

**Sample Output**:

The sorted three-dimensional array is: 0 1 2 3 1 2 3 4 2 3 4 5 3 4 5 6

4 5 6 7 5 6 7 8 6 7 8 9 7 8 9 10

|  |  |  |  |
| --- | --- | --- | --- |
|  | Test Case 1 | Test Case 2 | Test Case 3 |
| Input | x = 1, y = 4, z = 3 | x = 3, y = 2, z = 5 | x = 2, y = 3, z = 4 |
| Output | Thesorted three-dimensional array is: 0 1 2 1 2 3 2 3 4 3 4 5 | The sorted three-dimensional array is: 0 1 2 3 4 1 2 3 4 5 2 3 4 5 6 3 4 5 6 7 4 5 6 7 8  5 6 7 8 9 6 7 8 9 10 7 8 9 10 11 | The sorted three-dimensional array is: 0 1 2 3 1 2 3 4 2 3 4 5 3 4 5 6  4 5 6 7 5 6 7 8 6 7 8 9 7 8 9 10 |

**Solution:**

#include <stdio.h>

#include <stdlib.h>

int\*\*\* allocateMemory(int x, int y, int z) {

int\*\*\* arr = (int\*\*\*)malloc(x \* sizeof(int\*\*));

for (int i = 0; i < x; i++) {

arr[i] = (int\*\*)malloc(y \* sizeof(int\*));

for (int j = 0; j < y; j++) {

arr[i][j] = (int\*)malloc(z \* sizeof(int));

}

}

return arr;

}

void initializeElements(int\*\*\* arr, int x, int y, int z) {

for (int i = 0; i < x; i++) {

for (int j = 0; j < y; j++) {

for (int k = 0; k < z; k++) {

arr[i][j][k] = i + j + k;

}

}

}

}

void sortElements(int\*\*\* arr, int x, int y, int z) {

for (int i = 0; i < x; i++) {

for (int j = 0; j < y; j++) {

for (int k = 0; k < z - 1; k++) {

for (int l = 0; l < z - k - 1; l++) {

if (arr[i][j][l] > arr[i][j][l + 1]) {

int temp = arr[i][j][l];

arr[i][j][l] = arr[i][j][l + 1];

arr[i][j][l + 1] = temp;

}

}

}

}

}

}

int main() {

int x, y, z;

printf("Enter the number of x: ");

scanf("%d", &x);

printf("Enter the number of y: ");

scanf("%d", &y);

printf("Enter the number of z: ");

scanf("%d", &z);

int\*\*\* arr = allocateMemory(x, y, z);

initializeElements(arr, x, y, z);

sortElements(arr, x, y, z);

printf("The sorted three-dimensional array is: \n");

for (int i = 0; i < x; i++) {

for (int j = 0; j < y; j++) {

for (int k = 0; k < z; k++) {

printf("%d ", arr[i][j][k]);

}

printf("\n");

}

printf("\n");

}

return 0;

}