**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 1. What is the purpose of the rename() function in C file handling?

a) to delete a file

b) to rename a file

c) to create a new file

d) to open a file

Answer: b) to rename a file

Q2 1. What is the purpose of the ftell() function in C file handling?

a) to return the current position of the file pointer

b) to return the size of the file

c) to return the last error code

d) to return the current date and time

Answer: a) to return the current position of the file pointer

Q3 1. What is the purpose of the fseek() function in C file handling?

a) to move the file pointer to a specific location

b) to close a file

c) to read a line of text from a file

d) to write data to a file

Answer: a) to move the file pointer to a specific location

Q4 1. Which of the following is not a valid operation on a union in C?

a. Assigning a value to one member and reading from another member

b. Assigning a value to one member and reading from the same member

c. Assigning a value to all members at the same time

d. None of the above

Answer: c

Q5 1. What is the syntax for declaring an enumeration in C with named constants?

a. enum {constant1, constant2, constant3};

b. enum {constant1=1, constant2=2, constant3=3};

c. enum {1=constant1, 2=constant2, 3=constant3};

d. enum (constant1, constant2, constant3);

Answer: b

Q6 1. Which of the following headers do you need to include for file handling in C?

a) stdio.h

b) file.h

c) input.h

d) output.h

Answer: a) stdio.h

Q7 1. Which of the following is an example of a function that returns a struct in C?

A) int add(int a, int b);

B) float multiply(float a, float b);

C) struct point create\_point(int x, int y);

D) None of the above

Answer: C) struct point create\_point(int x, int y);

Q8 1. Which of the following is a user-defined data type in C?

a. int

b. float

c. double

d. enum

Answer: d

Q9 1. 1. What is the purpose of a helper function in recursion?

a) It is used to make the code shorter

b) It is used to make the code faster

c) It is used to avoid using global variables

d) None of the above

Answer: c) It is used to avoid using global variables

Q10 What is the maximum number of recursive calls that can be made in a program?

* 1. It depends on the size of the stack
  2. It depends on the size of the heap
  3. It is unlimited
  4. None of the above

Answer: a) It depends on the size of the stack

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

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

Q11 What is the output of the following recursive function when called with an argument of 3?

int mystery(int n) {

if (n <= 0) {

return 0;

}

return mystery(n-1) \* 2 + 1;

}

A. 1

B. 3

C. 7

D. 15

Q12 What is the output of the following code?

#include <stdio.h>

struct point {

int x;

int y;

};

void printPoint(struct point \*p) {

printf("(%d,%d)\n", p->x, p->y);

}

int main() {

struct point p1 = { 5, 10 };

struct point \*p2 = &p1;

p2->x = 15;

p2->y = 20;

printPoint(&p1);

return 0;

}

A. (5,10)

B. (15,20)

C. (5,20)

D. Error

Correct answer: B

Q13 What is the purpose of the . operator when used with a structure variable?

a. To access the address of the structure variable

b. To access a member of the structure variable

c. To allocate memory for the structure variable

d. To free memory for the structure variable

Correct answer: b

Q14 What is the output of the following code snippet?

#include <stdio.h>

enum fruits { APPLE, ORANGE, MANGO };

int main() {

enum fruits f = APPLE;

if(f > ORANGE) printf("Yes");

else printf("No");

return 0;

}

a) Yes

b) No

c) Compiler error

d) Runtime error

Answer: b

Q15 Which of the following is true about passing structures as function arguments in C?

a) Structures cannot be passed as arguments to functions.

b) Structures can only be passed as pointers to functions.

c) Structures can be passed as values or pointers to functions.

d) Structures can only be passed as arrays to functions.

Correct answer: c

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

Q16. John is a software developer working on a new project that requires him to write a C program to find the nth Fibonacci number using recursion. Help him to get the solution.

**Input:**

**Constraints:**

0<= n <= 30

**Output:**

**print nth term of**  Fibonacci sequence(0,1,1,2,3,5………)

Sample test Cases

|  |  |  |
| --- | --- | --- |
|  | Input | Output |
| STC1 | 0 | 0 |
| STC2 | 5 | 5 |

**Solution 16:**

#include <stdio.h>

int fibonacci(int num) {

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

return num;

} else {

return fibonacci(num - 1) + fibonacci(num - 2);

}

}

int main() {

int num;

scanf("%d", &num);

printf("%d\n",fibonacci(num));

return 0;

}

Test Cases

|  |  |  |
| --- | --- | --- |
|  | Input | Output |
| TC1 | 10 | 5510 |
| TC2 | 15 | 610 |
| TC3 | 30 | 832040 |
| TC4 | 1 | 1 |
| TC5 | 0 | 0 |

Q17. Create a union named "var" that includes an integer n, a float f, and a character c. Write a program to input data for each member of the union and display their values.

**Input:**

**Constraints:**

1<=**i**<=10000

1.00 <= **f**<=10000.00

char will contain upper and lower case alphabets.

**Output:**

**print 3 lines containing values of n,f and c**

Sample test Cases

|  |  |  |
| --- | --- | --- |
|  | Input | Output |
| STC1 | 20  30.5  a | 20  30.5  a |
| STC2 | 5  5.5  b | 5  5.5  b |

**Solution 17:**

#include<stdio.h>

union var {

int i;

float f;

char c;

};

int main() {

union var v;

scanf("%d", &v.i);

printf("%d\n", v.i);

scanf("%f", &v.f);

printf("%f\n", v.f);

scanf(" %c", &v.c);

printf("%c\n", v.c);

return 0;

}

Test Cases

|  |  |  |
| --- | --- | --- |
|  | Input | Output |
| TC1 | 10  10.10  a | 10  10.10  a |
| TC2 | 10000  10191.1  b | 10000  10191.1  b |
| TC3 | 6766  787.99  d | 6766  787.99  d |
| TC4 | 7887  67.88  A | 7887  67.88  A |
| TC5 | 999  9.99  Q | 999  9.99  Q |

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

Q18 **Given an array of N non-negative integers arr[] representing an elevation map where the width of each bar is 1, compute how much water it is able to trap after raining.**

**Sample Input**:

int arr[] = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

**Sample Output**:

The maximum amount of water that can be trapped: 0

|  |  |  |  |
| --- | --- | --- | --- |
|  | Test Case 1 | Test Case 2 | Test Case 3 |
| Input | int arr[] = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 }; int n = 10; | int arr[] = { 3, 1, 2, 4, 0, 1, 3, 2, 1, 2, 1 }; int n = 11; | int arr[] = { 0, 1, 0, 2, 1, 0, 1, 3, 2, 1, 2, 1 }; int n = 12; |
| Output | The maximum amount of water that can be trapped: 0 | The maximum amount of water that can be trapped: 13 | The maximum amount of water that can be trapped: 6 |

**Solution:**

#include <stdio.h>

// Creating MACRO for finding the maximum number

#define max(x, y) (((x) > (y)) ? (x) : (y))

// Creating MACRO for finding the minimum number

#define min(x, y) (((x) < (y)) ? (x) : (y))

// Function to return the maximum

// water that can be stored

int maxWater(int arr[], int n)

{

// To store the maximum water

int res = 0;

// For every element of the array

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

// Find the maximum element on its left

int left = arr[i];

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

left = max(left, arr[j]);

}

// Find the maximum element on its left

int right = arr[i];

for (int j = i + 1; j < n; j++) {

right = max(right, arr[j]);

}

// Update the result (maximum water)

res = res + (min(left, right) - arr[i]);

}

// Return the maximum water

return res;

}

int main()

{

int arr[] = { 0, 1, 0, 2, 1, 0, 1, 3, 2, 1, 2, 1 };

int n = sizeof(arr) / sizeof(arr[0]);

printf("%d", maxWater(arr, n));

return 0;

}