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

**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 recursion in C programming?

a) It is a function that calls itself

b) It is a function that uses loops

c) It is a function that uses switch case

d) None of the above

Answer: a) It is a function that calls itself

Q2 Which of the following is true about recursion?

a) It is faster than iteration

b) It is slower than iteration

c) It takes more memory than iteration

d) It takes less memory than iteration

Answer: c) It takes more memory than iteration

Q3 1. Which of the following is the base case in recursion?

a) The condition that is checked in every iteration

b) The condition that is checked only once

c) The condition that is not checked

d) None of the above

Answer: b) The condition that is checked only once

Q4 1. Which of the following is a user-defined data type in C?

a) int

b) float

c) double

d) struct

Answer: d) struct

Q5 1. What is the size of a struct with no members in C?

a) 0

b) 1

c) 2

d) 4

Answer: a) 0

Q6 1. Which operator is used to access members of a struct in C?

a) .

b) ->

c) &

d) \*

Answer: a) .

Q7 1. What is a union in C?

a) A data structure that allows you to store different data types in the same memory location

b) A collection of related data items

c) A set of rules for how data is stored in memory

d) None of the above

Answer: a

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. 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

Q10 1. What is the default mode when opening a file in C?

a) read mode

b) write mode

c) append mode

d) binary mode

Answer: a) read mode

**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?

void printNum(int n) {

if (n < 0) {

return;

}

printf("%d ", n);

printNum(n-1);

}

A. 0 1 2

B. 1 2 3

C. 3 2 1

D. 3 2 1 0

Q12 What is the output of the following code?

#include <stdio.h>

struct person {

char name[20];

int age;

float salary;

};

int main() {

struct person p = { "John", 25, 2500.50 };

printf("%s %d %.2f\n", p.name, p.age, p.salary);

return 0;

}

A. John 25 2500.50

B. John 25 2500

C. John 2500 25.50

D. Error

Correct answer: A

Q13 What does the typedef keyword do when used with a structure?

a. Defines a new type based on an existing structure type

b. Declares a new structure type with a new name

c. Assigns a value to a structure member

d. Initializes a structure variable

Correct answer: a

Q14 What is the output of the following code snippet?

#include <stdio.h>

enum colors { RED, GREEN, BLUE };

int main() {

enum colors c = BLUE;

printf("%d", c);

return 0;

}

a) 0

b) 1

c) 2

d) 3

Answer: c

Q15 What is the output of the following program?

#include <stdio.h>

#include <string.h>

union data {

int i;

float f;

char str[20];

};

int main() {

union data d;

d.i = 10;

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

d.f = 3.14;

printf("d.f = %f\n", d.f);

strcpy(d.str, "Hello World");

printf("d.str = %s\n", d.str);

return 0;

}

a) d.i = 10, d.f = 3.14, d.str = "Hello World"

b) d.i = 10, d.f = 3.14, d.str = garbage value

c) Compiler error

d) None of the above

Correct answer: a

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

Q16. You are transporting n numbers boxes through a tunnel, where each box is parallelepiped, and is characterized by its length, width and height.

The height of the tunnel feet and the width can be assumed to be infinite. A box can be carried through the tunnel only if its height is strictly less than the tunnel's height h. Find the volume of each box that can be successfully transported to the other end of the tunnel. Note: Boxes cannot be rotated.

Input: take two integer n and h.

n line have length, width and height of nth -1 box.

**constraints**:

1 <=n,h,length,width and height,height <= 100

Sample test Cases

|  |  |  |
| --- | --- | --- |
|  | Input | Output |
| STC1 | 4 41  5 5 5  1 2 40  10 5 41  7 2 42  Explanation: given 4 boxes and tunnel height is 41.  box0 has dimensions 5,5,5 as length,width,height.  box1 has dimension 1,2,40 as length,width,height.  box2 has dimension 10,5,41 as length,width,height.  box3 has dimensions 7,2,42 as length,width,height.  box 0 and 1 has height less than 41, hence printing their volumes as 125 and 80. | 125  80 |
| STC2 | 3 8  2 3 4  8 6 8  3 4 6  box 0 and 2 has heights less than tunnel height. hence they can be carried out though it. | 24  72 |

**Solution 16:**

#include <stdio.h>

#include <stdlib.h>

#define MAX\_HEIGHT 41

struct box

{

int length;

int width;

int height;

};

typedef struct box box;

int get\_volume(box b) {

int volume;

volume=(b.length)\*(b.width)\*(b.height);

return volume;

}

int is\_lower\_than\_max\_height(box b) {

int result = b.height < MAX\_HEIGHT ? 1:0;

return result;

}

int main()

{

int n,ma;

scanf("%d%d", &n,&MAX\_HEIGHT);

box \*boxes = malloc(n \* sizeof(box));

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

scanf("%d%d%d", &boxes[i].length, &boxes[i].width, &boxes[i].height);

}

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

if (is\_lower\_than\_max\_height(boxes[i])) {

printf("%d\n", get\_volume(boxes[i]));

}

}

return 0;

}

Test Cases

|  |  |  |
| --- | --- | --- |
|  | Input | Output |
| TC1 | 10  67 71 48  48 83 91  11 61 43  83 93 14  98 12 97  74 11 60  48 87 27  52 7 41  32 92 64  43 78 26 | 228336  28853  108066  112752  14924  87204 |
| TC2 | 20  28 7 39  79 69 5  82 67 90  37 83 91  31 97 46  14 13 53  61 59 72  10 28 38  25 16 65  8 16 91  27 68 83  17 55 86  75 20 44  92 93 44  70 12 5  37 21 10  96 19 79  11 83 20  55 19 80  45 67 89 | 7644  27255  494460  279461  138322  9646  259128  10640  26000  11648  152388  80410  66000  376464  4200  7770  144096  18260  83600  268335 |
| TC3 | 19 73  40 75 51  100 10 16  8 14 42  61 39 94  10 70 66  28 40 17  33 89 96  56 13 98  12 17 86  63 63 37  7 74 67  91 52 18  93 71 2  90 29 7  69 36 81  86 1 13  54 27 80  63 69 66  58 85 63 | 153000  16000  4704  46200  19040  146853  34706  85176  13206  18270  1118  286902  310590 |
| TC4 | 40  60 27 87  52 93 97  85 6 84  77 34 24  97 2 68  34 78 3  45 47 96  80 95 65  69 71 4 | 62832  7956  19596 |
| TC5 | 17 50  2 74 62  93 3 64  68 61 100  43 66 63  89 60 7  47 46 22  80 63 42  67 93 87  40 24 34  89 59 19  24 93 72  69 34 65  72 97 48  57 86 78  66 87 30  19 81 70  81 77 66 | 37380  47564  211680  32640  99769  335232  172260 |

Q17. Imagine you're a mathematician studying the properties of numbers. You come across a unique number n and its reverse r, such that when the number is raised to the power of its reverse, a fascinating result occurs. You have given a number and its reverse. You have to determine what the result of this calculation would be?

As answers can be very large, print the result modulo 10^9 + 7.

Input:

first line contain n, r.

**constraints**:

1 <= N <= 10^9

Output:

print n to the power r.

Sample test Cases

|  |  |  |
| --- | --- | --- |
|  | Input | Output |
| STC1 | 10 1  Explanation: The reverse of 10 is 1  and after raising power of 10 by 1  we get 10 which gives remainder as  10 by dividing 1000000007. | 10 |
| STC2 | 12 21  Explanation: The reverse of 12 is 21  and after raising power of 12 by 21  hence pow(12,21)%(10^9 + 7) = 864354781 | 864354781 |

**Solution 17:**

#include <stdio.h>

# define mod 1000000007

long long power(int N,int R)

{

if(R==0) return 1;

long long temp = power(N,R/2);

temp=(temp\*temp)%mod;

if(R%2==1)

{

return (temp\*N)%mod;

}

return temp;

}

int main()

{

long long N,R;

scanf("%ld%ld",&N,&R);

long long ans =power(N,R);

printf("%ld",ans);

}

Test Cases

|  |  |  |
| --- | --- | --- |
|  | Input | Output |
| TC1 | 23 32 | 929916350 |
| TC2 | 100 1 | 100 |
| TC3 | 9876 6789 | 418657441 |
| TC4 | 9001 1009 | 302734433 |
| TC5 | 99999999 99999999 | 139597207 |

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

**Q18 Imagine you are a librarian tasked with sorting a large collection of books in ascending order by book number.To accomplish this, you decide to use a selection sort algorithm implemented through recursion. Starting with the first book, you compare it to each subsequent book and swap their positions if necessary, continuing this process until the entire collection is sorted. By using recursion, you are able to break down the sorting process into smaller, more manageable steps, which helps to simplify the overall task of organizing the library's book collection.**

**Sample Input:**

**Enter the size of the list: 5**

**Enter the elements in list: 55,66,2,88,44**

**Sample Output:**

**The sorted book list in ascending order is**

**2 44 55 66 88**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | **44**  **88**  **66**  **102**  **5**  **7** | **502**  **433**  **958**  **107**  **534**  **789**  **124**  **428** | **1005**  **905**  **267** |
| **Output** | **5 7 44 66 88 102** | **107 124 428 433 502 534 789 958** | **267 905 1005** |

**Solution:**

**#include <stdio.h>**

**void selection(int [], int, int, int, int);**

**int main()**

**{**

**int list[30], size, temp, i, j;**

**printf("Enter the size of the list: ");**

**scanf("%d", &size);**

**printf("Enter the elements in list:\n");**

**for (i = 0; i < size; i++)**

**{**

**scanf("%d", &list[i]);**

**}**

**selection(list, 0, 0, size, 1);**

**printf("The sorted list in ascending order is\n");**

**for (i = 0; i < size; i++)**

**{**

**printf("%d ", list[i]);**

**}**

**return 0;**

**}**

**void selection(int list[], int i, int j, int size, int flag)**

**{**

**int temp;**

**if (i < size - 1)**

**{**

**if (flag)**

**{**

**j = i + 1;**

**}**

**if (j < size)**

**{**

**if (list[i] > list[j])**

**{**

**temp = list[i];**

**list[i] = list[j];**

**list[j] = temp;**

**}**

**selection(list, i, j + 1, size, 0);**

**}**

**selection(list, i + 1, 0, size, 1);**

**}**

**}**