1. Write a program in C to find the factorial of a given number using pointers.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Private Test cases used for evaluation** | **Input** | **Expected Output** | **Actual Output** | **Status** |
| Test Case 1 | 15 | The Factorial of 15 is: 1307674368000 | The Factorial of 15 is: 1307674368000\n | Passed |

#include <stdio.h>

void findFact(int, long int\*);

int main()

{

long int fact; //factorial of the number

int num1;

scanf("%d",&num1); //The number is taken from test data

findFact(num1, &fact);

printf("The Factorial of %d is: %ld\n",num1, fact);

return 0;

}

void findFact(int n, long int \*f)

{

int i;

\*f =1;

for(i=1;i<=n;i++)

\*f=\*f\*i;

}

1. Write a C program to print the Record of the Student Merit wise. Here a structure variable is defined which contains student rollno, name and score.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Private Test cases used for evaluation** | **Input** | **Expected Output** | **Actual Output** | **Status** |
| Test Case 1 | 4  1  Pradip  900  2  Asutosh  600  3  Santosh  750  4  Sandip  500 | The Merit List is :\n  1 Pradip 900\n  3 Santosh 750\n  2 Asutosh 600\n  4 Sandip 500 | The Merit List is :\n  1 Pradip 900\n  3 Santosh 750\n  2 Asutosh 600\n  4 Sandip 500\n | Passed |

#include<stdio.h>

struct student

{

int rollno;

char name[20];

int score;

};

void main()

{

struct student s[20];

int i, n;

scanf("%d" ,&n); //No. of Students taken from test data

// Roll no., Name and Score of n students are taken from test data

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

{

scanf("%d", &s[i].rollno);

scanf("%s", s[i].name);

scanf("%d", &s[i].score);

}

//Complete the program so that merit list is printed in descending order

struct student temp;

int j;

for(i=0;i<n-1;i++)

{

for(j=0;j<n-1;j++)

{

if(s[j].score<s[j+1].score)

{

temp=s[j];

s[j]=s[j+1];

s[j+1]=temp;

}

}

}

printf("The Merit List is :\n");

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

{

printf("%d", s[i].rollno);

printf(" %s", s[i].name);

printf(" %d\n", s[i].score);

}

}

1. Write a C program to store n elements using Dynamic Memory Allocation - calloc() and find the Largest element.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Private Test cases used for evaluation** | **Input** | **Expected Output** | **Actual Output** | **Status** |
| Test Case 1 | 6  68.90  34.79  35.86  94.98  40.06  88.70 | Largest element = 94.98 | Largest element = 94.98\n | Passed |

#include <stdio.h>

#include <stdlib.h>

int main()

{

int n;

float \*element;

scanf("%d", &n); //Total number of elements

// Allocate the memory for 'n' number of elements.

//Then take the elements as input from test data

// Use the printf statement as:

// printf("Largest element = %.2f\n", \*element);

element = (float\*) calloc(n, sizeof(float));

if(element == NULL)

{

printf("Error!!! memory not allocated.");

exit(0);

}

// Stores the number entered by the user.

int i;

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

{

scanf("%f", element + i);

}

// find the largest

for(i = 1; i < n; ++i)

{

if(\*element < \*(element + i))

\*element = \*(element + i);

}

printf("Largest element = %.2f\n", \*element);

return 0;

}

1. Write a C program to add two distance given as input in feet and inches.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Private Test cases used for evaluation** | **Input** | **Expected Output** | **Actual Output** | **Status** |
| Test Case 1 | 5  11  9  1 | Sum of distances = 15 feet 0 inches | Sum of distances = 15 feet 0 inches | Passed |

#include<stdio.h>

struct Distance{

int feet;

int inch;

}d1,d2,sum;

int main()

{

//Enter 1st distance

scanf("%d",&d1.feet);

scanf("%d",&d1.inch);

//Enter 2nd distance

scanf("%d",&d2.feet);

scanf("%d",&d2.inch);

scanf("%d",&d2.feet);

scanf("%d",&d2.inch);

sum.feet=d1.feet+d2.feet;

sum.inch=d1.inch+d2.inch;

if(sum.inch>=12)

{

sum.inch=sum.inch-12;

++sum.feet;

}

printf("Sum of distances = %d feet %d inches",sum.feet,sum.inch);

return 0;

}