Rajalakshmi Engineering College

Name: SONASREE RP
Email: 240701521@rajalakshmi.edu.in
Roll no: 240701521
Phone: 7305340666
Branch: REC
Department: I CSE FE
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 0_Pointers

Attempt : 1
Total Mark : 5
Marks Obtained : 5

Section 1 : Coding

1. Problem Statement

Daniel is working on a project that involves analyzing data stored in float arrays. He needs to determine whether a given float array contains only positive numbers.

To achieve this, he needs a program that can accurately evaluate the contents of float arrays using malloc().

Input Format

The first line of input consists of an integer N, representing the size of the array.

The second line consists of N space-separated float values, representing the elements of the array.

Output Format

If all the array elements are positive, print "All elements are positive."

If the array contains at least one positive element, print "At least one element is positive."

If there are no positive elements in the array, print "No positive elements in the array."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
50.0 -2.3 3.7 -4.8 5.2
Output: At least one element is positive.

Answer

```
// You are using GCC
#include<stdio.h>
#include<stdlib.h>

int main() {
    int N;

    scanf("%d",&N);

    float *arr = (float *)malloc(N * sizeof(float));
    for(int i=0; i<N; i++){
        scanf("%f",&arr[i]);
    }

    int allpositive = 1;
    int atleastonepositive = 0;

    for(int i=0; i<N; i++){
        if (arr[i]>0){
            atleastonepositive = 1;
        } else {
            allpositive = 0;
        }
```

```
    }
    if(allpositive){
        printf("All elements are positive.\n");
    } else if (atleastonepositive) {
        printf("At least one element is positive.\n");
    } else {
        printf("No positive elements in the array.");
    }
    free(arr);
    return 0;

}
```

Status : Correct **Marks : 1/1**

2. Problem Statement

Rajwinder wants a program to determine retirement details for a person based on their age.

Create a program that uses a structure called Person to hold the age as an attribute with a pointer.

If the age is under 18, display "Invalid".If the age is 65 or older, print "Already retired!".Otherwise, calculate and output the retirement year, remaining years, and remaining days until retirement.

Note: Age 65 is considered as retirement age. Assume the current year as 2023 and there are 365 days per year for calculation.

Input Format

The input consists of an integer representing the person's age.

Output Format

If the age is under 18, the output displays "Invalid" and terminates.

If the age is 65 or older, the output displays "Already retired!" and terminates.

Otherwise, the output displays the following.

1. The first line displays "Retirement Year: " followed by an integer representing the retirement year.
2. The second line displays "Remaining Years: " followed by an integer representing the remaining years left for retirement.
3. The third line displays "Remaining Days: " followed by an integer representing the remaining days left for retirement.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 43

Output: Retirement Year: 2045
Remaining Years: 22
Remaining Days: 8030

Answer

```
// You are using GCC
#include<stdio.h>
struct Person{
    int age;
}p;
int main()
{
    scanf("%d",&p.age);
    if(p.age>=65){
        printf("Already retired!");
        return 0;
    }
    if(p.age<18){
        printf("Invalid");
        return 0;
    }
    p.age=65-p.age;
    printf("Retirement Year: %d\nRemaining Years: %d\nRemaining Days: %d",p.age
+2023,p.age,p.age*365);
}
```

Status : Correct ***Marks : 1/1***

3. Problem Statement

Raj wants to create a program using pointers and a structure named Employee to manage employee information.

He seeks your assistance to input the employee's name, salary, and hours worked. Implement a salary increase based on hours worked, and calculate the final salary. Calculate the total salary for 30 days. Display the results of the final and total salary.

Salary increase criteria:

If hours worked >= 12, the increase is Rs. 150.00.If hours worked >= 10, but less than 12, the increase is Rs. 100.00.If hours worked >= 8, but less than 10, the increase is Rs. 50.00.If hours worked < 8, there is no increase.

Input Format

The first line of input consists of a string, representing the Employee's name.

The second line consists of a double-point number, representing the Employee's current salary.

The third line consists of an integer, representing the number of hours worked by the employee.

Output Format

The first line of output prints "Final Salary: Rs. " followed by a double value, representing the final salary, rounded off to two decimal places.

The second line prints "Total Salary: Rs. " followed by a double value, representing the total salary for 30 days, rounded off to two decimal places.

Refer to the sample outputs for formatting specifications.

Sample Test Case

Input: Akil
3000.00
6

Output: Final Salary: Rs. 3000.00
Total Salary: Rs. 90000.00

Answer

```
#include <stdio.h>
#include <string.h>
struct Employee{
    char name[50];
    float salary;
    int hoursworked;
};

void calculateSalary(struct Employee *emp){
    if(emp ->hoursworked >=12){
        emp->salary +=150;
    }else if(emp ->hoursworked >= 10){
        emp->salary += 100.0;
    }else if (emp->hoursworked >= 8){
        emp->salary += 50.0;
    }
}

int main(){
    struct Employee emp;
    scanf("%s",emp.name);
    scanf("%f",&emp.salary);
    scanf("%d",&emp.hoursworked);
    calculateSalary(&emp);
    float totalSalary = emp.salary * 30;
    printf("Final Salary: Rs. %.2f\n",emp.salary);
    printf("Total Salary: Rs. %.2f\n",totalSalary);
    return 0;
}
```

Status : Correct ***Marks : 1/1***

4. Problem Statement

Sam is developing a program for analyzing daily temperature fluctuations. Users input the number of days, followed by daily temperature values whose memory is allocated using malloc.

The program calculates and displays the following:

The absolute temperature changes between consecutive days (The first value remains the same).The average temperature of adjacent days.

This allows users to gain insights into daily temperature variations for better analysis.

For Example,

Let us assume the temperature for 3 days as 25.5, 28.0, and 23.5.

The absolute differences:

Day 1: (N/A, as there is no previous day) = 25.50Day 2: abs(28.0 - 25.5) = 2.50Day 2: abs(23.5 - 28.0) = 4.50

The average temperatures:

Day 1: (N/A, as there is no previous day) = 25.50Day 2: (25.5 + 23.5) / 2.0 = 24.50Day 3: (N/A, as there is no next day) = 23.50

Input Format

The first line consists of an integer N, representing the number of days.

The second line consists of N space-separated float values, representing the temperature values for N days.

Output Format

The first line displays the absolute temperature change for N days as float values, rounded to two decimal places, separated by a space.

The second line displays the average temperature for N days as float values, rounded to two decimal places, separated by a space.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
25.5 28.0 23.5

Output: 25.50 2.50 4.50
25.50 24.50 23.50

Answer

```
#include <stdio.h>
#include <stdlib.h>

int main() {
    int a;
    float d;
    scanf("%d",&a);
    float*p;

   p = (float*)malloc(a*sizeof(float));

    for(int i=0;i<a;i++)
    {
        scanf("%f",&p[i]);
    }

    for(int i=0;i<a;i++)
    {
        if(i == 0)
        {
        printf("%.2f ",p[i]);
        }
        else
        {
            d=p[i]-p[i-1];
            if(d<0)
            {
                d=-d;
            }
            printf("%.2f ",d);
        }
    }
    printf("\n");
    for(int i=0;i<a;i++)
    {
        if(i==0)
        {
            printf("%.2f ",p[i]);
```

```
        }
        else
        {
            if(i==a-1)
            {
                printf("%.2f ",p[i]);
            }
            else
            {
                d=(p[i+1]+p[i-1])/2;
                p[i]=d;
                printf("%.2f ",d);
            }
        }
    }
    return 0;
}
```

Status : Correct ***Marks : 1/1***

5. Problem Statement

Ria is a mathematician who loves exploring combinatorics. She is working on a project that involves calculating permutations.

Ria wants to create a program that takes the values of n and r as input and calculates the permutations of n elements taken r at a time.

Write a program using pointers and a function calculatePermutations that, given the values of n and r, calculates and prints the permutations of n elements taken r at a time.

Permutation: n! / (n - r)!

Input Format

The first line consists of an integer n, representing the total number of elements.

The second line consists of an integer r, representing the number of elements to be taken at a time.

Output Format

The output prints the result of the permutation.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
3
Output: 24

Answer

```
// You are using GCC
#include<stdio.h>
long long calculatePermutations(int n,int r){
    long long result = 1;
    for(int i=0;i<r;i++){
        result *=(n-i);
    }
    return result;
}


int main() {
    int n,r;

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

    if(r>n){
        printf("0\n");
        return 0;
    }

    printf("%lld\n", calculatePermutations(n,r));

    return 0;
```

}

Status : Correct ***Marks : 1/1***