Ahmed Shafique, Haiqa Qaiser 2021-EE-313, 2022-EE-255

Data Structures and Algorithms

Lab 1 & 2: Fundamentals

Q1): Write separate function of each of the following:

Functions:

a. Sum of Arithmetic Series:

```
double arithmeticSeriesSum(double a, double d, int n)
{
    double sum = (n/2) * (2 * a + (n - 1) * d);
    return sum;
}
```

b. Sum of Geometric Series:

```
double geometric_series(double a, double r, int n) {
    if (r > 1) {
        return a * (pow(r, n) - 1) / (r - 1);
    } else if (r < 1) {
        return a * (1 - pow(r, n)) / (1 - r);
    } else {
        cout << "The common ratio cannot be 1." << endl;
        return -1;
    }
}
```

c. Sum of infinite geometric Series:

```
double geometric_infinite_series(double a, double r) {
    if (r < 1) {
        return a / (1 - r);
    } else if (r > 1) {
        return a / (r - 1);
    } else {
        cout << "The common ratio cannot be 1." << endl;
        return -1;
    }
}
```

d. A function that return a structure Z:

```
void z()

{

    geometric_infinite_series(double a, double r);

    geometric_series(double a, double r, int n);

}
```

Q#02: Write a program that uses the functions to develop in Q1 in order to:

1. Asks the user which function he/she wants to calculate? (3)

2. Accepts from the user appropriate inputs, note that each function mentioned above may require a different set of inputs.

3. Calls the requisite function to calculate the formula, prints the results and exits.

```
int main()

{

    int choice;
```

```
    cout<<"Hello! What you want to calculate from the following?
type(1 or 2 or 3)";cin>>choice;
    double a, d, r;
    int n;
    if (choice==1)
    {
        cout<<"Enter the value of a : ";cin>>a;
        cout<<"Enter the value of d : ";cin>>d;
        cout<<"Enter the value of n : ";cin>>n;
        double sum1 = arithmeticSeriesSum(a,d,n);
        cout<<"Sum of Arithmatic Series: "<<sum1<<endl;
    }
    else if (choice ==2)
    {
        cout<<"Enter the value of a : ";cin>>a;
        cout<<"Enter the value of r : ";cin>>r;
        cout<<"Enter the value of n : ";cin>>n;
        double sum2 = geometric_series(a, r, n);
        cout<<"Sum of Arithmatic Series: "<<sum2<<endl;
    }
    else
    {
        cout<<"Enter the value of a : ";cin>>a;
        cout<<"Enter the value of r : ";cin>>r;
        double sum3 = geometric_infinite_series(a,r);
        cout<<"Sum of Arithmatic Series: "<<sum3<<endl;
    }
    return 0;
}
```

Q#03: Write a program that:

1. Declare a structure named ‘student.’ Each student should contain a name and marks of three courses, namely, {Math, Chem, Physics}.

2. Make an array of structures, where the size of the array is 10.

3. Initialize the names and marks of these 10 students via the user.

4. Call out a function named ‘calculate_average’ which accepts this array of structures and evaluates and prints the respective averages of Math, Chemistry and Physics.

5. Call out a function names ‘bubble_sort’ which accepts this array of structures and sorts and prints the student list (i.e., their names and marks) as per their names.

Code:

```
struct student {
    string name;
    double chem;
    double phy;
    double math;
};
int length = 0;
void initial(int n, struct student **data_arr) {
    cout << "----------- Loading ----------" << endl;
    for (int i = 0; i < n; i++) {
        cout << "[" << i + 1 << "]: " << "Enter the name of Student:
";
        cin >> data_arr[i]->name;
        cout << endl;
        cout << "Enter the chemistry marks: ";
        cin >> data_arr[i]->chem;
        cout << "Enter the physics marks: ";
        cin >> data_arr[i]->phy;
        cout << "Enter the math marks: ";
        cin >> data_arr[i]->math;
        cout << endl;
    }
    cout << "---------- Please Wait ----------" << endl;
    length = length + 1;
    cout << endl;
    return;
}

void calculate_average() {
    // Your implementation for calculating average goes here
    return;
}

bool compareByName(const student* a, const student* b) {
    return a->name < b->name;
}
```

```
void sort(int n, struct student **data_arr) {
    cout << "---------- Alphabetized order ----------" << endl;
    sort(data_arr, data_arr + n, compareByName);
    return;
}


void printing(int n, struct student **data_arr) {
    cout << "---------- Loading ----------" << endl;
    cout << endl << "---------- Recorded Data ----------" << endl;
    for (int i = 0; i < n; i++) {
        cout << "[" << i + 1 << "]: " << "Name: " << data_arr[i]-
>name
            << " Chemistry: " << data_arr[i]->chem
            << " Physics: " << data_arr[i]->phy
            << " Math: " << data_arr[i]->math << endl;
        double avg = (data_arr[i]->chem + data_arr[i]->phy +
data_arr[i]->math) / 3.0;
        cout << " Average: " << avg << endl;
    }
    cout << endl;
    length = length + 1;
}


int main() {
    int n;
    cout << "Enter the number of student's data that you want to
store in structure: ";
    cin >> n;
    cout << endl;
    student* arr[n]; // array of structure
    student* ptr_arr[n]; // pointer array points to structure.
    for (int i = 0; i < n; i++) {
        ptr_arr[i] = new student; // new pointer structure & new
multiples dynamic memory allocations.
    }
    cout << endl;
    initial(n, ptr_arr); // calling with original ptr_arr[] instead
of data_arr[].
    sort(n, ptr_arr);
    printing(n, ptr_arr);
```

```
        cout << "Length of Function: " << length << endl;
        for (int i = 0; i < n; i++) {
            delete ptr_arr[i];
        }
        return 0;
    }
```

Output:

```
Enter the number of student's data that you want to store in structure: 2


---------- Loading ----------
[1]: Enter the name of Student: Ahmed

Enter the chemistry marks: 99
Enter the physics marks: 98
Enter the math marks: 99
```

Fig: 01 Q3

```
[2]: Enter the name of Student: Haiqa

Enter the chemistry marks: 98
Enter the physics marks: 99
Enter the math marks: 98
```

Fig: 02 Q3

```
[1]: Name: Ahmed Chemistry: 99 Physics: 98 Math: 99
 Average: 98.6667
[2]: Name: Haiqa Chemistry: 98 Physics: 99 Math: 98
 Average: 98.3333
```

Fig: 03 Q3

```
Hello! What you want to calculate from the following? type(1 or 2 or 3)2
Enter the value of a : 25
Enter the value of r : 4
Enter the value of n : 8
Sum of Arithmatic Series: 546125
```

Fig: 04 Q1 & Q2

```
Hello! What you want to calculate from the following? type(1 or 2 or 3)2
Enter the value of a : 4
Enter the value of r : 6
Enter the value of n : 8
Sum of Geometric Series: 1.34369e+006
```

Fig: 05 Q1 & Q2

```
Hello! What you want to calculate from the following? type(1 or 2 or 3)3
Enter the value of a : 5
Enter the value of r : 9
Sum of Infinite Geometric Series: 0.625
```

Fig: 06 Q1 & Q2

Methodology:

This C++ program calculates the sum of either an arithmetic series, geometric series, or infinite geometric series based on user input. It defines functions for each calculation: `arithmeticSeriesSum` for arithmetic series, `geometric_series` for finite geometric series, and `geometric_infinite_series` for infinite geometric series. The `main` function prompts the user to choose the type of series and input necessary values, then computes and displays the corresponding sum.

This C++ program manages student data by defining a `student` struct with attributes for name, chemistry marks, physics marks, and math marks. It implements functions to initialize data, calculate averages, sort data alphabetically by name, and print recorded data with their averages. The `main` function prompts the user to input the number of students, then records and displays their data. Finally, it deallocates memory used by dynamically allocated structures.