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

Data Structures and Algorithms

Lab 5: Matrices-I

Q1): Write a program that:

1 A function that initializes $Xm \times n$ and $Y \times k$, where {m, n, l, k} are user defined and also its 10 contents.

2 A function that evaluates $Z = X + Y$. 5

3 A function that evaluates $Z = X - Y$. 5

4 A function called 'print' that simply prints the answer.

Code:

```
struct element
{
    int r,c, *p;
};

void initialize_matrix(int row, int col, int *ptr)
{
    int k;
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < col; j++)
        {
            cin>>k;
            *(ptr + i*col + j) = k;
        }
    }
    return;
}


element addition_matrces(int r1, int c1, int r2, int c2, int *p1, int *p2)
{
    element matrix;
    try
    {
        if ((r1==r2) && (c1==c2))
        {
           cout<<"Processing addition of matrices"<<endl;
            matrix.c = c1;
            matrix.r = r1;
            matrix.p = new int [r1*c1];
            int a,b,c;
            for (int i = 0; i < r1; i++)
            {
                for (int j = 0; j < c1; j++)
                {
                    a= *(p1 + i*c1 + j);
                    b = *(p2 + i*c1 + j);
                    c = a + b;
                    *(matrix.p + i*c1 + j) = c;
                }
            }
        }
        else throw(c1);
    }
    catch(...)
```

```
    {
        cout<<"Addition is not possible at this time"<<endl;
        matrix.c = 0;
        matrix.r = 0;
        matrix.p = NULL;
    }
    return matrix;
}

element substraction_matrces(int r1, int c1, int r2, int c2, int *p1, int
*p2)
{
    element matrix;
    try
    {
        if ((r1==r2) && (c1==c2))
        {
            cout<<"Processing subtraction of matrices"<<endl;
            matrix.c = c1;
            matrix.r = r1;
            matrix.p = new int [r1*c1];
            int a,b,c;
            for (int i = 0; i < r1; i++)
            {
                for (int j = 0; j < c1; j++)
                {
                    a= *(p1 + i*c1 + j);
                    b = *(p2 + i*c1 + j);
                    c = a - b;
                    *(matrix.p + i*c1 + j) = c;
                }
            }
        }
        else throw(c1);
    }
    catch(...)
    {
        cout<<"substraction is not possible at this time"<<endl;
        matrix.c = 0;
        matrix.r = 0;
        matrix.p = NULL;
    }
    return matrix;
}

void printing_matrix(int row, int col, int *ptr, string name)
{
    cout<<"Dear "<<name<<"! "<<"Matrix X has following contents: "<<endl;
    for (int i = 0; i < row; i++)
    {
        cout << "[";
        for (int j = 0; j < col; j++)
        {
            cout << " " << *ptr; // Corrected dereferencing here
            ptr = ptr + 1;
        }
        cout << " ]" << endl;
    }
    return;
}

void add_printing_matrix(int row, int col, int *ptr, string name)
{
    cout<<"Dear "<<name<<"! "<<"Z = X + Y is shown below:"<<endl;
    for (int i = 0; i < row; i++)
```

```
    {
        cout << "[";
        for (int j = 0; j < col; j++)
        {
            cout << " " << *ptr; // Corrected dereferencing here
            ptr = ptr + 1;
        }
        cout << " ]" << endl;
    }
    return;
}

void sub_printing_matrix(int row, int col, int *ptr, string name)
{
    cout<<"Dear "<<name<<"! "<<"Z = X - Y is shown below:"<<endl;
    for (int i = 0; i < row; i++)
    {
        cout << "[";
        for (int j = 0; j < col; j++)
        {
            cout << " " << *ptr; // Corrected dereferencing here
            ptr = ptr + 1;
        }
        cout << " ]" << endl;
    }
    return;
}


int main()
{
    int r1, r2, c1, c2, *p1, *p2;
    string user;
    cout<<"Enter name of current user?: ";cin>>user;cout<<endl;
    cout<<"Dear "<<user<<"! "<<"Tell me what are the dimensions of X?:
"<<endl;
    cin>>r1;
    cin>>c1;
    p1 = new int [r1*c1];
    cout<<"Dear "<<user<<"! "<<"Enter the contents of X:"<<endl;
    initialize_matrix(r1,c1,p1);
    printing_matrix(r1,c1,p1, user);

    cout<<"Dear "<<user<<"! "<<"Tell me what are the dimensions of Y?:
"<<endl;
    cin>>r2;
    cin>>c2;
    p2 = new int [r2*c2];
    cout<<"Dear "<<user<<"! "<<"Enter the contents of Y:"<<endl;
    initialize_matrix(r2,c2,p2);
    printing_matrix(r2,c2,p2,user);

    element matrix1, matrix2;
    matrix1 = addition_matrces(r1,c1,r2,c2,p1,p2);
    add_printing_matrix(matrix1.r, matrix1.c, matrix1.p, user);

    matrix2 = substraction_matrces(r1,c1,r2,c2,p1,p2);
    sub_printing_matrix(matrix2.r, matrix2.c, matrix2.p, user);
    delete [] p1;
    delete [] p2;
    return 0;
}
```

Output:

```
Enter name of current user?: Ahmed

Dear Ahmed! Tell me what are the dimensions of X?:
2
2
Dear Ahmed! Enter the contents of X:
4
4
4
4
Dear Ahmed! Matrix X has following contents:
[ 4 4 ]
[ 4 4 ]
Dear Ahmed! Tell me what are the dimensions of Y?:
2
2
```

Fig: 01

```
Dear Ahmed! Enter the contents of Y:
3
2
2
3
Dear Ahmed! Matrix X has following contents:
[ 3 2 ]
[ 2 3 ]
Processing addition of matrices
Dear Ahmed! Z = X + Y is shown below:
[ 7 6 ]
[ 6 7 ]
Processing subtraction of matrices
Dear Ahmed! Z = X - Y is shown below:
[ 1 2 ]
[ 2 1 ]
```

Fig: 02

Methodology:

This code performs matrix addition and subtraction operations based on user input. It first prompts the user to enter their name and the dimensions of two matrices, X and Y. Then, it initializes memory space for the matrices and prompts the user to input their contents. After that, it calculates the addition and subtraction of the matrices, ensuring that they have the same dimensions.

For each operation, it dynamically allocates memory for the resultant matrix and performs the arithmetic calculations. It also handles exceptions if the dimensions of the matrices are incompatible for addition or subtraction. Finally, it prints the contents of the input matrices and the result matrices, along with a personalized message addressed to the user.

Memory allocated for the matrices is properly deallocated at the end to avoid memory leaks. Overall, the code is structured to facilitate matrix operations while providing clear feedback to the user throughout the process.