PROGRAM NO: 1 DATE: 10-08-2023

**AIM**

To find minimum and maximum number present in an array.

**ALGORITHM**

1. START
2. Initialize min and max variable with first element of the array.
3. Start traversing the array from 1st element to last element (nth)
4. Compare each value with min and max variables.
5. If the value is higher than max, replace the max value.
6. If the value is lower than min, replace the min value.
7. Outside the for loop, print the min and max values.
8. STOP

**CODE**

// Find maximum and minimum element in an array.

#include <iostream>

using namespace std;

void findMinMax(int arr[], int size) {

int i, min = arr[0], max = arr[0];

for (i=1; i<size; i++) {

if (arr[i] < min) {

min = arr[i];

}

if (arr[i] > max) {

max = arr[i];

}

};

cout<< "Minimum value is : " << min <<endl;

cout<< "Maximum value is : " << max <<endl;

cout<<endl;

}

int main() {

int size, i;

cout<< "Enter size of array: ";

cin>> size;

int arr[size];

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

cout<< "Enter element " <<i + 1 <<" : ";

cin>>arr[i];

}

cout<<endl;

findMinMax(arr, size);

cout<< "--------- Author ----------------" <<endl;

cout<< "Ali Izzath Shazin" <<endl;

cout<< "220071601028" <<endl;

cout<< "B. Tech CSE A" <<endl;

return 0;

}

**OUTPUT**

Enter size of array: 5

Enter element 1 : 5

Enter element 2 : -2

Enter element 3 : -3

Enter element 4 : 2

Enter element 5 : 6

Minimum value is : -3

Maximum value is : 6

--------- Author ----------------

Ali Izzath Shazin

220071601028

B. Tech CSE A

PROGRAM NO: 2 DATE: 10-08-2023

**AIM**

To find consecutive letters if they appear in the given character array.

**ALGORITHM**

1. START
2. Initialize variables i, flag (int) and text (string)
3. Traverse through the char array starting from 1st index to last.
4. Inside for loop, check if ASCII value of current character and last character are consecutive.
5. If yes, and flag is 0, print the last and current character, and set flag to 1.
6. If flag is 1, print only the current character.
7. If ASCII value of current character and last character are not consecutive, and if flag is 1.
8. Print an endline and set flag to 0.
9. STOP

**CODE**

*// Find consecutive letters if they appear in the given character array.*

#include <iostream>

using namespace std;

int main() {

    int i, flag=0;

    string text;

    cout<< "Enter the text to check : ";

    cin>> text;

    int length = text.length();

    for (i=1; i<length; i++) {

        if (text[i] - 1 == text[i-1] || text[i] - text[i-1] - 1 == 32 || ftext[i-1] - text[i] + 1 == 32) {

            if (flag == 0) {

                cout<<text[i - 1] << text[i];

                flag = 1;

            } else {

                cout<< text[i];

            }

        } else {

            if (flag == 1) {

                cout<< endl;

                flag = 0;

            }

        }

    }

    cout<<endl;

    cout<< "--------- Author ----------------" <<endl;

    cout<< "Ali Izzath Shazin" <<endl;

    cout<< "220071601028" <<endl;

    cout<< "B. Tech CSE A" <<endl;

    return 0;

}

**OUTPUT**

Enter the text to check :AbcDhlmNquvWz

AbcD

lmN

uvW

--------- Author ----------------

Ali Izzath Shazin

220071601028

B. Tech CSE A

PROGRAM NO: 3 DATE: 17-08-2023

**AIM**

To implement insert,delete and traverse operations on linear array.

**ALGORITHM**

1. START
2. For Displaying, initialize a for loop from i=0 to i=length of array (exclusive).
3. Print every element inside the array.
4. For Inserting, initialize a for loop from i=last index to i=index to insert to.
5. Inside loop, replace (i + 1)th element with ith element.
6. Outside loop, insert new element to the given index.
7. Return the incremented length pointer.
8. For Deleting, initialize a for loop from i=index given to i=length of array - 1 (exclusive)
9. Inside loop, replace ith element with (i + 1)th element.
10. Return the decremented length pointer.
11. STOP

**CODE**

#include <iostream>

using namespace std;

void Display(int arr[], int &n, int &size) {

    cout<<endl;

    cout<< "[";

    for (int i=0; i<n; i++) {

        if (i == n - 1) cout<<arr[i];

        else cout<<arr[i] << ", ";

    }

    cout<< "] Size = " << size <<endl<<endl;

}

void Insert(int arr[], int &n, int size, int index, int item) {

    if (n == size) {

        cout<< "Array overflow!" <<endl<<endl;

        return;

    }

    if (index > n || index < 0) {

        cout<< "Array out of bound" <<endl<<endl;

        return;

    }

    for (int i=n-1; i>= index; i--) {

        arr[i + 1] = arr[i];

    }

    arr[index] = item;

    n++;

}

void Delete(int arr[], int &n, int index) {

    if (index > n - 1 || index < 0) {

        cout<<endl;

        cout<< "Array out of bound" <<endl;

        return;

    }

    for (int i=index; i<n-1; i++) {

        arr[i] = arr[i + 1];

    }

    n--;

}

int main() {

    int size, n, i, choice, temp1, temp2;

    cout<< "Enter size of array: ";

    cin>> size;

    int \*arr = new int[size];

    while (true) {

        cout<< "Enter number of elements to insert: ";

        cin>> n;

        if (n > size) {

            cout<< "Number of elements to insert should be lesser than or

sequal to the given size!" <<endl;

            continue;

        }

        break;

    }

    for (i=0; i<n; i++) {

        cout<< "Enter (" <<i + 1 <<")th element: ";

        cin>>arr[i];

    }

    while (true) {

        cout<< "----------- Array Operation -----------" <<endl;

        cout<< "1. Display Array." <<endl;

        cout<< "2. Insert Array." <<endl;

        cout<< "3. Delete Array." <<endl;

        cout<< "4. Exit." <<endl;

        cout<< "Enter choice: ";

        cin>> choice;

        if (choice == 1) {

            Display(arr, n, size);

        } else if (choice == 2) {

            cout<<endl<< "Enter element to insert: ";

            cin>> temp1;

            cout<< "Enter index to insert to: ";

            cin>> temp2;

            cout<< endl;

            Insert(arr, n, size, temp2, temp1);

        } else if (choice == 3) {

            cout<<endl<< "Enter index to delete: ";

            cin>> temp1;

            Delete(arr, n, temp1);

            cout<< endl;

        } else if (choice == 4) {

            cout<<endl<< "Thank you. Exiting.." <<endl;

cout<< "--------- Author ----------------" <<endl;

    cout<< "Ali Izzath Shazin" <<endl;

    cout<< "220071601028" <<endl;

    cout<< "B. Tech CSE A" <<endl;

            return 0;

        } else {

            cout<<endl<< "Invalid choice" <<endl<< endl;

            continue;

        }

    }

    return 0;}

**OUTPUT**

Enter size of array: 5

Enter number of elements to insert: 3

Enter (1)th element: 1

Enter (2)th element: 2

Enter (3)th element: 3

----------- Array Operation -----------

1. Display Array.

2. Insert Array.

3. Delete Array.

4. Exit.

Enter choice: 1

[1, 2, 3] Size = 5

----------- Array Operation -----------

1. Display Array.

2. Insert Array.

3. Delete Array.

4. Exit.

Enter choice: 2

Enter element to insert: 10

Enter index to insert to: 0

----------- Array Operation -----------

1. Display Array.

2. Insert Array.

3. Delete Array.

4. Exit.

Enter choice: 1

[10, 1, 2, 3] Size = 5

----------- Array Operation -----------

1. Display Array.

2. Insert Array.

3. Delete Array.

4. Exit.

Enter choice: 3

Enter index to delete: 1

----------- Array Operation -----------

1. Display Array.

2. Insert Array.

3. Delete Array.

4. Exit.

Enter choice: 1

[10, 2, 3] Size = 5

----------- Array Operation -----------

1. Display Array.

2. Insert Array.

3. Delete Array.

4. Exit.

Enter choice: 4

Thank you. Exiting..

--------- Author ----------------

Ali Izzath Shazin

220071601028

B. Tech CSE A

PROGRAM NO: 4 DATE: 24-08-2023

**AIM**

To implement addition, subtraction and multiplication of matrix.

**PROGRAM**

1. START
2. For storing matrix as pointer to pointer variable.
3. Firstly, initialize a pointer to pointer, and assign it with a pointer array of row size.
4. Loop through the pointer array and create array of column size in each index.
5. For addition and subtraction.
6. Declare two for loops. One traversing from i=0 to i=row size. Second one traversing from j=0 to j = col size.
7. Create a third zero matrix and assign the sum to each indices.
8. Finally, return the third matrix.
9. For multiplication of matrix.
10. Declare three for loops, One traversing from i=0 to i=row size of mat1. Second from j=0 to j=col size of mat2. Third one traversing from k=0 to k=col size of mat1 or row size of mat2
11. Then inside all three loops, mat3[i][j] += mat1[i][k] \* mat2[k][j];
12. STOP

**CODE**

#include <iostream>

using namespace std;

class Exception {

    public:

    virtual string msg() = 0;

};

class InvalidSize: public Exception {

    public:

    string msg() {

        return "Size should be greater than zero.";

    }

};

class OperationFailed: public Exception {

    public:

    string msg() {

        return "Both matrix should be of same size.";

    }

};

class MultiplicationFailed: public Exception {

    public:

    string msg() {

        return "Number of columns in the 1st matrix must be equal to the rows in the 2nd matrix.";

    }

};

class Matrix {

    int \_rows, \_cols;

    int \*\*mat;

    Matrix doAddOrSub(Matrix obj, int instance) {

        if (this->\_rows != obj.\_rows || this->\_cols != obj.\_cols) throw OperationFailed();

        Matrix returnMat = Matrix(this->\_rows, this->\_cols);

        returnMat.initializeMatrix();

        for (int i=0; i<this->\_rows; i++) {

            for (int j=0; j<this->\_cols; j++) {

                if (instance == 0) {

                    returnMat.mat[i][j] = this->mat[i][j] + obj.mat[i][j];

                } else if (instance == 1) {

                    returnMat.mat[i][j] = this->mat[i][j] - obj.mat[i][j];

                }

            }

        }

        return returnMat;

    }

    void initializeMatrix() {

        this->mat = new int\*[this->\_rows];

        for(int i = 0; i < this->\_rows; i++) {

            this->mat[i] = new int[this->\_cols];

        }

    }

    public:

        Matrix(int rows, int cols) {

            if (rows <= 0 || cols <= 0) throw InvalidSize();

            this->\_rows = rows;

            this->\_cols = cols;

            this->initializeMatrix();

        }

        Matrix operator+(Matrix obj) {

            return doAddOrSub(obj, 0);

        }

        Matrix operator-(Matrix obj) {

            return doAddOrSub(obj, 1);

        }

        Matrix operator\*(Matrix obj) {

            if (this->\_cols != obj.\_rows) throw MultiplicationFailed();

            Matrix returnMat = Matrix(this->\_rows, obj.\_cols);

            returnMat.initializeMatrix();

            for(int i=0; i<this->\_rows; i++) {

                for(int j=0; j<obj.\_cols; j++) {

                    returnMat.mat[i][j]=0;

                    for(int k=0; k<this->\_cols; k++) {

                        returnMat.mat[i][j] += this->mat[i][k] \* obj.mat[k][j];

                    }

                }

            }

            return returnMat;

        }

        void askElements() {

            for (int i=0; i<this->\_rows; i++) {

                for (int j=0; j<this->\_cols; j++) {

                    cout << "Enter element (" << i+1 << ", " << j+1 << "): ";

                    cin >> this->mat[i][j];

                }

            }

        }

        void display() {

            for (int i=0; i<this->\_rows; i++) {

                for (int j=0; j<this->\_cols; j++) {

                    cout << this->mat[i][j] << "\t";

                }

                cout << endl;

            }

            cout << "(" << this->\_rows << ", " << this->\_cols << ")" << endl;

        }

        int rows() {

            return this->\_rows;

        }

        int cols() {

            return this->\_cols;

        }

};

ostream& operator<<(ostream &out, Matrix obj) {

    obj.display();

    return out;

}

int main() {

    int mat1\_r, mat1\_c, mat2\_r, mat2\_c;

    cout << "Enter matrix 1 rows and columns: ";

    cin >> mat1\_r >> mat1\_c;

    Matrix mat1 = Matrix(mat1\_r, mat1\_c);

    cout << "Enter matrix 1 elements: " << endl;

    mat1.askElements();

    cout << mat1 << endl;

    cout << "Enter matrix 2 rows and columns: ";

    cin >> mat2\_r >> mat2\_c;

    Matrix mat2 = Matrix(mat2\_r, mat2\_c);

    cout << "Enter matrix 2 elements: " << endl;

    mat2.askElements();

    cout << mat2 << endl;

    try {

        cout << "Matrix 1 + matrix 2" << endl;

        Matrix mat3 = mat1 + mat2;

        cout << mat3 << endl;

    } catch(Exception &e) {

        cout << "Addition Failed: " << e.msg() << endl;

    }

    try {

        cout << "Matrix 1 - matrix 2" << endl;

        cout << mat1 - mat2 << endl;

    } catch(Exception &e) {

        cout << "Subtraction Failed: " << e.msg() << endl;

    }

    try {

        cout << "Matrix 1 \* matrix 2" << endl;

        cout << mat1 \* mat2 << endl;

    } catch(Exception &e) {

        cout << "Multiplication Failed: " << e.msg() << endl;

    }

    return 0;

}

**OUTPUT**

Enter matrix 1 rows and columns: 2 2

Enter matrix 1 elements:

Enter element (1, 1): 1

Enter element (1, 2): 2

Enter element (2, 1): 3

Enter element (2, 2): 4

1 2

3 4

(2, 2)

Enter matrix 2 rows and columns: 2 2

Enter matrix 2 elements:

Enter element (1, 1): 1

Enter element (1, 2): 2

Enter element (2, 1): 3

Enter element (2, 2): 4

1 2

3 4

(2, 2)

Matrix 1 + matrix 2

2 4

6 8

(2, 2)

Matrix 1 - matrix 2

0 0

0 0

(2, 2)

Matrix 1 \* matrix 2

7 10

15 22

(2, 2)