```
// A safe array example.
#include <iostream>
#include <cstdlib>
#include<string.h>
using namespace std;


class atype{
    int ncols;
    int nrows;
    int *dynamicArray;
    public:

    atype(){
        nrows=0;
        ncols=0;;
        dynamicArray=NULL;


    }
    //constructor
    atype(int row, int col){
        nrows=row;
        ncols=col;
        int size=nrows*ncols;
        dynamicArray = new int[size];



    }

    //destructor
    ~atype(){

        delete [] dynamicArray;


    }

    //user inserting elements in 2d array
    void fillArray()
      {
          int size=nrows*ncols;
          for (int in=0;in<size;in++){

                        int value;
                        cout<<"enter values";
                        cin>>value;

                        dynamicArray[in] = value;


                }

      }


    //bound checking-safe array implementation
    int &operator ()(int i, int j){
        if(i<0 || i> nrows-1 || j<0 || j>  ncols-1 ) {
```

```
        cout << "Boundary Error\n";
        exit(1);

    }
    long offset=(i*ncols)+j;
return dynamicArray[offset];
}

//copy constructor
atype(const atype& rhs)
{

     nrows = rhs.nrows;
     ncols = rhs.ncols;
     int size=nrows*ncols;
     dynamicArray = new int[size];

    memcpy(dynamicArray,rhs.dynamicArray, sizeof(int)*nrows*ncols);




}

//assignment operator overloading
atype& operator=(const atype& rhs)
  {
       if (this == &rhs)
       return *this;


    delete [] dynamicArray;

    nrows = rhs.nrows;
    ncols = rhs.ncols;
    int size=nrows*ncols;
    dynamicArray = new int[size];

    memcpy(dynamicArray,rhs.dynamicArray, sizeof(int)*nrows*ncols);




    return *this;
  }


//not equal to operator overloading
bool  operator!=(const atype& arr){
    if (this->nrows != arr.nrows || this->ncols != arr.ncols)
            return false;

      int size = nrows * ncols;



  // now check the contents
      for (int i = 0; i < size; i++) {
```

```
                if (dynamicArray[i] != arr.dynamicArray[i]){

                    return false;
                }
                  else{
                      return true;
                  }
          }



    }

};




int main()
{
      int columns;
      int rows;
      cout<<"enter number of rows and cols"<<endl;
      cin>>rows>>columns;
      atype ob1(rows,columns);

      ob1.fillArray();
      cout<<endl<<"----------------"<<endl;
      atype ob2=ob1;
      atype ob3(2,2);

      ob3.fillArray();

      //ob2.fillArray();
      cout << ob1(1,1) << endl;

      cout<<ob3(1,1)<<endl; //checking bounds of array
      bool b=ob1!=ob3;

      cout<<b;


      return 0;
}
```