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


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

    atype(){

        ncols=0;;
        dynamicArray = new int[ncols];



    }
    //constructor
    atype(int col){

        ncols=col;
        dynamicArray = new int[ncols];


    }

    //destructor
    ~atype(){

       delete [] dynamicArray;


    }

    //user inserting elements in 2d array
    void fillArray()
      {

          for (int in=0;in<ncols;++in){
                    int value;
                    cout<<"enter value";
                    cin>>value;
                    dynamicArray[in] = value;
          }

      }


    //bound checking-safe array implementation
    int &operator [](int i){
        if(i<0 || i>  ncols-1 ) {
            cout << "Boundary Error\n";
            exit(1);

        }
    return dynamicArray[i];
    }
```

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

      ncols = rhs.ncols;

      dynamicArray = new int[ncols];


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

 }

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

     delete[] dynamicArray;


     ncols = rhs.ncols;
     dynamicArray = new int[ncols];
     memcpy(dynamicArray,rhs.dynamicArray, sizeof(int)*ncols);


      return *this;
   }

atype& operator!=(const atype& rhs){
    for (int i=0;i<ncols;i++){
        if(dynamicArray[i]!=rhs.dynamicArray[i]){
            cout<<"not equal";
            break;
        }




     }


}



friend istream &operator>> (istream &input, const atype &array)
 {
  int cols;

    cols=array.ncols;

  for (int i = 0; i < cols; i++) {

         input >> array.dynamicArray[i];

  }
    return input;
 }
```

```
};



int main()
{
	int columns;

	cout<<"enter cols"<<endl;
	cin>>columns;
	atype ob1(columns);

	ob1.fillArray();
	cin>>ob1; //if not using fill array option, this will call friend
function declared for taking input
	atype ob2=ob1;
	atype ob3;

	ob3=ob1;
	cout << ob1[1] << endl;
	cout<<ob1[2]<<endl;	//checking bounds of array
	cout<<ob3[2];

	ob1!=ob3;


	return 0;
}
```