|  |
| --- |
| **NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES**  **CS 201–DATA STRUCTURES LAB**  **Lab Session 03** |
| **Instructors:** Mr. Faizan Yousuf, Ms. Safia, Ms. Maham Mobin |

## Outline

## Safe Array

## Jagged Array

## Safe Array:

## In C++, there is no check to determine whether the array index is out of bounds. During program execution, an out-of-bound array index can cause serious problems. Also, recall that in C++ the array index starts at 0. Safe array solves the out-of-bound array index problem and allows the user to begin the array index starting at any integer, positive or negative.

## Sample code:

|  |
| --- |
| #include<iostream>#include<fstream>#include <stdio.h>#include <string.h>#include<cstdlib>using namespace std;class ArrayIndexOutOfBoundException {public:ArrayIndexOutOfBoundException(): message ("Array Index Out of Bound") {}~ArrayIndexOutOfBoundException() { /\*delete message; \*/}const string what() const { return message; }private:char \*message;};class DynamicSafeArray{private:int \*Data;int size;public:DynamicSafeArray(){Data=NULL;size=0;}DynamicSafeArray(int n){size=n;Data= new int[size];memset(this->Data, 0, sizeof(int)\*size); }DynamicSafeArray(const DynamicSafeArray & rhs){this->size= rhs.size;this->Data= new int[size];memcpy(this->Data,rhs.Data,(sizeof(int)\*rhs.getSize()));}~DynamicSafeArray(){if(Data !=0){cout<<"bye--"<<endl;delete [] Data;Data=0;size=0; }}void ReSize(int nSize){if (size != nSize ){int \* temp= new int[size];for(int i=0; i<size ; i++){temp[i]= \*(Data+i);}delete[] Data;Data=0;Data = new int[nSize];memset(this->Data, 0, sizeof(int)\*nSize);for(int i=0; i<size ; i++){\*(Data+i)=temp[i];}size= nSize;delete [] temp;temp=0;}}unsigned int getSize() const {return size;}DynamicSafeArray& operator=( DynamicSafeArray & rhs){if (this != &rhs){int s=rhs.getSize();this->size=s;this->Data= new int[s];memcpy(this->Data,rhs.Data, sizeof(rhs.Data));}return (\*this);}//lvalint& operator[](unsigned int i){return \*(Data+i);}//rvalconst int& operator[](unsigned int i) const {return \*(Data+i);}friend istream& operator >> (istream& infile, DynamicSafeArray & rhs){for (int count=0;count<rhs.size;count++)infile>>rhs.Data[count];return infile;}friend ostream& operator << (ostream& outfile,DynamicSafeArray & rhs){for (int count=0;count<rhs.size;count++)outfile<<rhs.Data[count];return outfile;}};int main(){DynamicSafeArray DSA1(10);cout << "Enter an Array DSA1: ";cin >> DSA1;cout << DSA1;std::ofstream out\_file;out\_file.open("output.txt");out\_file << DSA1<< endl;out\_file.close();return 0;} |

**Example:**

|  |
| --- |
| #include <iostream>  #include <cstdlib>  using namespace std;  **const** **int** SIZE = 10;  template <**class** T> **class** MyType {  T a[SIZE];  **public**:  MyType() {  register **int** i;  **for**(i=0; i<SIZE; i++) a[i] = i;  }  T &operator[](**int** i){  **if**(i<0 || i> SIZE-1) {  cout << "\nIndex value of ";  cout << i << " is out-of-bounds.\n";  exit(1);  }  **return** a[i];  }  };  **int** main()  {  MyType<**int**> int ob;  MyType<**double**> doubleob;  cout << "Integer array: ";  **for**(**int** i=0; i<SIZE; i++)  intob[i] = i;  **for**(**int** i=0; i<SIZE; i++)  cout << intob[i] << " ";  cout << '\n';  cout << "Double array: ";  **for**(**int** i=0; i<SIZE; i++)  doubleob[i] = (**double**) i/3;  **for**(**int** i=0; i<SIZE; i++)  cout << doubleob[i] << " ";  cout << '\n';  intob[12] = 100;  **return** 0;  } |

**Output**:

* Integer array: 0 1 2 3 4 5 6 7 8 9
* Double array: 0 0.333333 0.666667 1 1.33333 1.66667 2 2.33333 2.66667 3
* Index value of 12 is out-of-bounds. In the program, pay special attention to this statement:

Int ob [12] = 100;

# It attempts to assign 100 to location 12 within int ob. But int ob is only 10 elements long! If this were a normal array, then a boundary overrun would occur. Fortunately, in this case, the attempt is intercepted by operator [] ( ) and the program is terminated before any damage can be done. (In actual practice, some sort of error-handling would be supplied to deal with the out-of-range condition; the program would not have to terminate.)

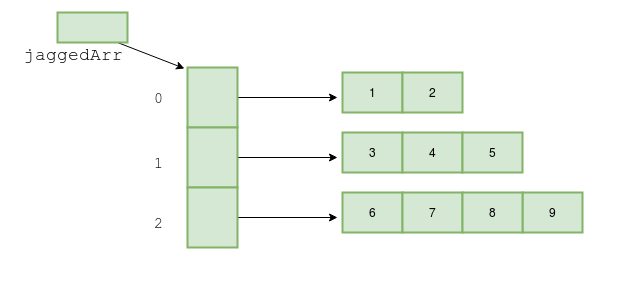
## Sample Pointer example:

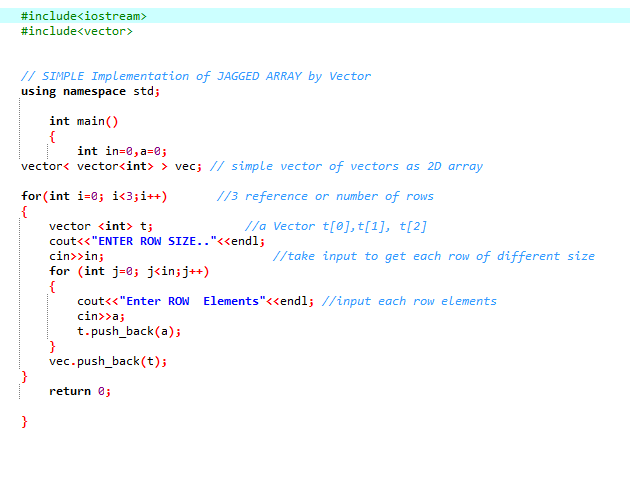
|  |
| --- |
| #include <iostream>  using namespace std;  /\* A point class of interger co-ordinates as point in 2D \*/  /\* it has got two data members abscissa (x) and ordinate (y) \*/  /\* this example class use dynamic memory for objects \*/    class Point2D{  private :  int \* itsX;  int \* itsY;  public:    /\* default constructor ------------------- \*/  /\* grab memory using new operator and initialize \*/  Point2D(){    itsX = new int;  itsY = new int;  \*itsX=\*itsY=0;  cout<< "Default Constructor" << endl;  }  /\* parametric constructor ------------------- \*/  /\* grab memory using new operator and initialize \*/  Point2D(int a, int b){    itsX = new int;  itsY = new int;  \*itsX=a;  \*itsY=b;  cout<< "Parametric Constructor" << endl;  }    /\* User define destructor ------------------- \*/  /\* lease memory using delete operator and set it to null \*/  ~Point2D()  {  if (itsX != 0) delete itsX; itsX=0;  if (itsY != 0) delete itsY; itsY=0;  cout<< "User-define destructor" << endl;  }  Point2D(const Point2D & rhs){    itsX = new int;  itsY = new int;  \*itsX=rhs.getX();  \*itsY=rhs.getY();  cout<< "Copy Constructor" << endl;  }    int getX() const { return \*itsX;}  int getY() const { return \*itsY;}    void setX(int a)  { itsX = new int;  \*itsX=a;  }    void setY(int b)  { itsY = new int;  \*itsY=b;  }  Point2D operator=( const Point2D& rhs){    cout<< "Assignment Operator" << endl;  if (this != &rhs)  {    delete itsX;  delete itsY;  itsX= new int;  \*itsX = rhs.getX();  itsY= new int;  \*itsY = rhs.getY();  }  return(\*this);    }  };  // Driver Program  int main(){    Point2D P1, P2(2,3);    cout << P1.getX() << P1.getY() << endl;  cout << P2.getX() << P2.getY() << endl;    P1=P2;    cout << P1.getX() << P1.getY() << endl;      return 0;  } |

# Jagged Array:

A jagged array is also a multi-dimensional array, comprising arrays of varying sizes as its elements (rows). It also referred ragged array.

Jagged array in memory:





# Task:

Write a program that will read n integers from the keyboard and place them in a jagged array as shown in the following diagram:

