Nahrin Sharna

Lab04

# Question 01

## Initial Test Plan:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test # | Valid / Invalid Data | Description of test | Input Value | Expected Output | Actual Output | Test Pass / Fail |
| 1 | Valid | Print() is called for vector a | Vector a, a.print() | Empty vector |  |  |
| 2 | Valid | Print() is called for vector b(3) to display initialized display | Vector b(3),  b.print() | [0 0 0] |  |  |
| 3 | Invalid | Set method is called for vector c where the pos value is less than valid range | Vector c(3),  c.set(0, -1) | Error messages |  |  |
| 4 | Valid | Set method is used to set values of c | Vector c(3),  c.set(1,0); | No error message |  |  |
| 5 | Valid | Set method is used to set values of c | Vector c(3),  c.set(2,1); | No error message |  |  |
| 6 | Valid | Set method is used to set values of c | Vector c(3),  c.set(3,2); | No error message |  |  |
|  | Invalid | Set method is used to set values of c | Vector c(3),  c.set(4,3); | Error message |  |  |
|  | Valid | Deep copy to initialize new object to the created object of same class | Vector c(3);  c.set(0,-1)  c.set(1,0);  c.set(2,1);  c.set(3,2);  c.set(4,3);  c.print();  Vector d(c);  d.print();  d.set(0,1);  d.print();  c.print(); | [1 2 3]  [1 2 3]  [1 0 3]  [1 2 3] |  |  |
|  | valid | Deep copy construction is being used to initialize new object of same class | Vector a;  Vector e(a);  e.print() | Empty vector |  |  |

## Source Code:

**Header File (Vector.h):**

#ifndef VECTOR\_H

#define VECTOR\_H

class Vector

{

public:

Vector(); // default constructor

Vector(int s); // makes size = s,

//allocates s space

// e.g. entries = new int[size],

// makes all entries 0

Vector(const Vector & other);

// copy constructor

// makes a deep copy

~Vector(); // default destructor

void print(); // Prints out the vector

void set(int val, int pos); // if 0 <=pos<size

// stores val at pos in entries

// otherwise

// error message

private:

int size; // sets the # of elements used

int \*entries; // point to array of integers with size entries

// e.g. entries = new int[size]

};

#endif

**Implementation File(Vector.cpp):**

#include<iostream>

#include "Vector.h"

using namespace std;

Vector::Vector() {

size = 0;

entries = new int[size];

}

Vector::Vector(int s) {

size = s;

entries = new int[size];

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

entries[i] = 0;

}

}

Vector::Vector(const Vector & other) {

size = other.size;

entries = new int[size];

for (int i = 0; i < other.size; i++) {

entries[i] = other.entries[i];

}

}

Vector::~Vector() {

delete[]entries;

}

void Vector::print() {

cout << "[";

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

cout << entries[i] << " ";

}

cout << "]";

cout << endl;

}

void Vector::set(int v, int p) {

// if 0 <=pos<size

if (p >= 0 && p < size) {

entries[p] = v; // stores val at pos in entries

} // otherwise

else

cout << "Error!" << endl; // error message

}

**Main File(Source.cpp):**

#include<iostream>

#include "Vector.h"

using namespace std;

int main() {

Vector a, b(3), c(3);

a.print(); // outputs []

b.print(); // outputs [ 0 0 0 ]

c.set(0, -1); // output error message

c.set(1, 0);

c.set(2, 1);

c.set(3, 2);

c.set(4, 3); // outputs error message

c.print(); // outputs [ 1 2 3 ]

Vector d(c);

d.print(); // outputs [ 1 2 3 ]

d.set(0, 1);

d.print(); // outputs [ 1 0 3 ]

c.print(); // outputs [ 1 2 3 ] proves deep copy \*/

Vector e(a);

e.print(); //output []

system("pause");

return 0;

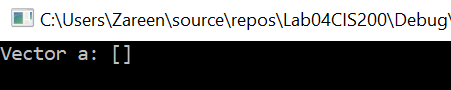
}

## Final Test Plan:

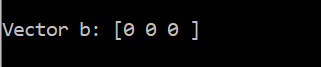
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test # | Valid / Invalid Data | Description of test | Input Value | Expected Output | Actual Output | Test Pass / Fail |
| 1 | Valid | Print() is called for vector a | Vector a, a.print() | Empty vector | [] | Pass |
| 2 | Valid | Print() is called for vector b(3) to display initialized display | Vector b(3),  b.print() | [0 0 0] | [0 0 0] | Pass |
| 3 | Invalid | Set method is called for vector c where the pos value is less than valid range | Vector c(3),  c.set(0, -1) | Error messages | “Error!Invalid input” | Pass |
| 4 | Valid | Set method is used to set values of c | Vector c(3),  c.set(1,0); | No error message | No error message | Pass |
| 5 | Valid | Set method is used to set values of c | Vector c(3),  c.set(2,1); | No error message | No error message | Pass |
| 6 | Valid | Set method is used to set values of c | Vector c(3),  c.set(3,2); | No error message | No error message | Pass |
| 7 | Invalid | Set method is used to set values of c | Vector c(3),  c.set(4,3); | Error message | “Error!Invalid input” | Pass |
| 8 | Valid | Deep copy to initialize new object to the created object of same class | Vector c(3);  c.set(0,-1)  c.set(1,0);  c.set(2,1);  c.set(3,2);  c.set(4,3);  c.print();  Vector d(c);  d.print();  d.set(0,1);  d.print();  c.print(); | [1 2 3]  [1 2 3]  [1 0 3]  [1 2 3] | [1 2 3]  [1 2 3]  [1 0 3]  [1 2 3] | Pass |
| 9. | valid | Deep copy construction is being used to initialize new object of same class | Vector a;  Vector e(a);  e.print() | Empty vector | [] | Pass |

## Screenshots:

**Test Case 1:**



**Test Case 2:**



**Test Case 3:**



**Test Case 4:**

**Nothing to display**

**Test Case 5:**

**Nothing to display**

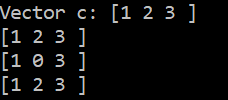
**Test Case 6:**

**Nothing to display**

**Test Case 7:**



**Test Case 8:**



**Test Case 9:**

