|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test | Description | Change code | Expected Output | Passed |
| 1 | Check constructor initialises object | Vector<int> testVec;  printIntVec(testVec); | Length = 0  Size = 0  NULL | Y |
| 2 | Check SetSize function | Vector<int> testVec;  testVec.SetSize(10);  printIntVec(testVec); | Length = 0  Size = 10  NOT NULL | Y |
| 3 | Check SetSize function with 0 | Vector<int> testVec;  testVec.SetSize(0); | Zero or less!!!  Length = 0  Size = 0  NULL | Y |
| 4 | Check copy function | Vector<int> testVec, testVec02; testVec.SetSize(10);  if(testVec02.CopyVec(testVec))  cout << "Vector copied" << endl; printIntVec(testVec); | Vector copied  Length = 0  Size = 10  NOT NULL | Y |
| 5 | Check contructor with size parameter | Vector<int> testVec(11);  printIntVec(testVec); | Length = 0  Size = 11  NOT NULL | Y |
| 6 | Check copy contructor | Vector<int> testVec(17);  for(int i = 0; i < 3; i ++)  testVec.AddItem(i);  Vector<int> testVec02(testVec);  printIntVec(testVec); | Length = 3  Size = 3  NOT NULL  Element 0 is: 0  Element 1 is: 1  Element 2 is: 2 | Y |
| 7 | Check AddItem adds item to end of Vector | Vector<int> testVec(3);  for(int i = 0; i < 3; i++)  testVec.AddItem(i);  printIntVec(testVec); | Length = 3  Size = 3  NOT NULL  Element 0 is: 0  Element 1 is: 1  Element 2 is: 2 | Y |
| 8 | Test overloaded operater [] | Vector<int> testVec(3);  for(int i = 0; i < 3; i++)  testVec.AddItem(i);  for(int i = 0; i < 3; i++)  testVec[i] = i + 10;  printIntVec(testVec); | Length = 3  Size = 3  NOT NULL  Element 0 is: 10  Element 1 is: 11  Element 2 is: 12 | Y |
| 9 | Test GetArray | Vector<int> testVec(3);  int \*testArr;  for(int i = 0; i < 3; i++)  testVec.AddItem(i);  testVec.GetArray(testArr);  printIntVec(testVec); | Length = 3  Size = 3  NOT NULL  Element 0 is: 0  Element 1 is: 1  Element 2 is: 2 |  |
| 10 | Test AddItem outside of range | Vector<int> testVec(2);  for(int i = 0; i < 4; i++)  {  if(testVec.PushBack(i))  cout << "Inside Range" << endl;  else  cout << "Outside Range" << endl;  }  printIntVec(testVec); | Inside Range  Inside Range  Outside Range  Outside Range  Length = 3  Size = 3  NOT NULL  Element 0 is: 0  Element 1 is: 1 | Y |
| 11 | Test SetArray | Vector<string> testVec(3);  string strArr[] = {"Apples", "Oranges", "Pears"};  if(testVec.SetArray(strArr, 3))  printIntVec(testVec); | Length = 3  Size = 3  NOT NULL  Element 0 is: Apples  Element 1 is: Oranges  Element 2 is: Pears | Y |
| 12 | Test GetLength and GetSize | Vector<int> testVec(3);  for(int i = 0; i < 2; i++)  testVec.PushBack(i);  cout << testVec.GetLength() << endl;  cout << testVec.GetSize() << endl; | 2  3 | Y |
| 13 | Tested with string object | Vector<string> testVec(3);  string testString = "String 0";  string emptyStr;  for(int i = 0; i < 3; i++)  {  testVec.PushBack(testString);  testVec.GetItem(emptyStr, i);  cout << emptyStr << i << endl;  } | String 00  String 01  String 02 | Y |
| 14 | Test assignment operator | Vector<string> testVec(3);  Vector<string> testVec2;  string testString = "String 0";  for(int i = 0; i < 3; i++)  {  testString = testString + to\_string(i);  testVec.AddItem(testString);  testString = "String 0";  }  testVec2 = testVec;  printIntVec(testVec2); | Length = 3  Size = 3  NOT NULL  Element 0 is: String 00  Element 1 is: String 01  Element 2 is: String 02 | Y |