1. **Inserting and removing elements from the middle of the vector is generally fast**

|  |  |
| --- | --- |
| **A** | True |
| **B** | False |

1. **What do arrays and linked lists have in common?**

|  |  |
| --- | --- |
| **A** | Nothing |
| **B** | Some features |
| **C** | Everything |

1. **What is the output?.**

*#include <iostream>*

*#include <algorithm>*

*#include <vector>*

*using namespace std;*

*void myFunction (int i)*

*{*

*cout << " " << 10\* i;*

*}*

*int main()*

*{*

*vector<int> myVector;*

*myVector.push\_back(10);*

*myVector.push\_back(20);*

*myVector.push\_back(30);*

*for\_each(myVector.begin(), myVector.end(), myFunction);*

*return 0;*

*}*

|  |  |
| --- | --- |
| **A** | 10 30 |
| **B** | 100 |
| **C** | 30 20 10 |
| **D** | 100 200 300 |

1. **What is the output?**

*#include <iostream>*

*#include <algorithm>*

*#include <vector>*

*using namespace std;*

*int main()*

*{*

*vector<int> myVector;*

*myVector.push\_back(10);*

*myVector.push\_back(20);*

*myVector.push\_back(30);*

*myVector.push\_back(40);*

*vector<int>::iterator it;*

*it = find(myVector.begin(), myVector.end(), 30);*

*++it;*

*cout << \*it << endl;*

*return 0;*

*}*

|  |  |
| --- | --- |
| **A** | 10 20 30 |
| **B** | 10203040 |
| **C** | 40 |
| **D** | 31 |

1. **Which of the following is true about Set?**
2. A set is a container that stores unique elements.
3. A setallows duplicate elements.
4. The elements of Set are sorted according to their values

|  |  |
| --- | --- |
| **A** | 1 and 2 |
| **B** | 1 and 3 |
| **C** | 2 and 3 |
| **D** | None of the mentioned |

1. **Which of the following can serve as random-access iterator?**

|  |  |
| --- | --- |
| **A** | Memory pointer |
| **B** | Object pointer |
| **C** | Class pointer |
| **D** | None of the mentioned |

1. **Output of following program?**

*#include <iostream>*

*#include <set>*

*using namespace std;*

*int main()*

*{*

*set<int> tst;*

*tst.insert(12);*

*tst.insert(21);*

*tst.insert(32);*

*tst.insert(31);*

*set<int> :: const\_iterator pos;*

*for(pos = tst.begin(); pos != tst.end(); ++pos)*

*cout << \*pos << ' ';*

*return 0;*

*}*

|  |  |
| --- | --- |
| **A** | 12 21 32 31 |
| **B** | 12 21 32 31 |
| **C** | 12 21 32 |
| **D** | 12 21 31 |

1. **By using which operator does point to next element is represent in iterator?**

|  |  |
| --- | --- |
| **A** | ++ |
| **B** | -- |
| **C** | +- |
| **D** | None of the mentioned |

1. **In which type of semantics does c++ implements iterator?**

|  |  |
| --- | --- |
| **A** | Memory |
| **B** | Size |
| **C** | Pointer |
| **D** | None of the mentioned |

1. **Output of following program?**

|  |
| --- |
|  |

*#include "list"*

*using namespace std;*

*int main(){*

*list<int> li;*

*for (int i = 0; i < 3; i++){*

*li.push\_back(i);*

*}*

*list<int>::iterator it = li.begin();*

*while (it!= li.end())*

*{*

*cout << \*it << " ";*

*it++;*

*}*

*return 0;*

*}* **Output:**

|  |  |
| --- | --- |
| **A** | 0 1 2 |
| **B** | 0 |
| **C** | 2 1 0 |
| **D** | 0 1 |

**Result:**

1B

2.A

2.D for\_each goes through the vector in order from the beginning to the endpoints specified.

3.C "find" returns 30 and incrementing the iterator makes it point to 40 (doesn't change the value, since it's a pointer).

5.B

6.b Explanation:Because of this, It can serve as any category of iterator.

7.b because when use Set, the elements are sorted according to their values

8.a

9.c

10.a