**Homework 9 컴퓨터공학부 202211390 최준원**

|  |
| --- |
| Q1 |
| Source Code |
| #include <iostream>  #include <fstream>  #include <vector>  #include <string>  #include <cassert>  using namespace std;  bool SearchFromFile(string str, int v);  // Implement SearchFromFile function  int main()  {  vector<int> nums = { 17, 72, 73 };  for (auto i = 0; i < nums.size(); i++)  {  int value = nums[i];  cout << "Search the integer to look for: " << value << endl;  // Printing the result  if (SearchFromFile("File1.txt", value))  {  cout << value << " is in the file. " << endl;  }  else  {  cout << value << " is not in the file!" << endl;  }  }  printf("\n Press Any Key to Terminate ...\n");  getchar();  return 0;  }  bool SearchFromFile(string str, int v) {  int data;  ifstream instream;  instream.open(str);  if (!instream.is\_open()) {  cout << "File cannot be opened." << endl;  assert(false);  }  for (int i = 1; i < 10; i++) {  instream >> data;  if (data == v) {  return true;  }  }  instream.close();  return false;  } |
| Screenshot |
| 텍스트, 스크린샷, 폰트이(가) 표시된 사진  자동 생성된 설명 |

|  |
| --- |
| Q2 |
| Code |
| #include <iostream>  #include <cassert>  #include <string>  using namespace std;  template <typename T>  struct Node  {  T data;  Node <T>\* next;  };  // Extend the List class  template <typename T>  class List  {  private:  Node <T>\* begin;  int count;  Node <T>\* makeNode(const T& value);  public:  List();  ~List();  void insert(int pos, const T& value);  void erase(int pos);  T& get(int pos) const;  void print() const;  void reversePrint() const;  void RPHelper(Node<T>\* head) const;  int size() const;  };  int main()  {  // Instantiation of a List object  List <int> list1;  // Insert six nodes in the list  list1.insert(0, 17);  list1.insert(1, 23);  list1.insert(2, 11);  list1.insert(3, 18);  // Printing the value of list in forward direction  cout << "Printing the list1" << endl;  list1.print();  cout << endl << endl;  // Printing the value of list in reverse direction  cout << "Printing the list1 in reverse order" << endl;  list1.reversePrint();  cout << endl << endl;  // Instantiation of a List object  List <string> list2;  // Insert six nodes in the list  list2.insert(0, "Michael");  list2.insert(1, "Jane");  list2.insert(2, "Sophie");  list2.insert(3, "Thomas");  list2.insert(4, "Rose");  list2.insert(5, "Richard");  // Printing the value of list in forward direction  cout << "Printing the list2" << endl;  list2.print();  cout << endl << endl;  // Printing the value of list in reverse direction  cout << "Printing the list2 in reverse order" << endl;  list2.reversePrint();  cout << endl << endl;  printf("\n Press Any Key to Terminate ...\n");  getchar();  return 0;  }  // Constructor  template <typename T>  List <T> ::List()  :begin(0), count(0)  {  }  // Destructor  template <typename T>  List <T> :: ~List()  {  Node <T>\* del = begin;  while (begin)  {  begin = begin -> next;  delete del;  del = begin;  }  }  // Insert member function  template <typename T>  void List <T> ::insert(int pos, const T& value)  {  if (pos < 0 || pos > count)  {  cout << "Error! The position is out of range." << endl;  return;  }  Node <T>\* add = makeNode(value);  if (pos == 0)  {  add->next = begin;  begin = add;  }  else  {  Node <T>\* cur = begin;  for (int i = 1; i < pos; i++)  {  cur = cur->next;  }  add->next = cur->next;  cur->next = add;  }  count++;  }  // MakeNode member function (private)  template <typename T>  Node <T>\* List <T> ::makeNode(const T& value)  {  Node <T>\* temp = new Node <T>;  temp->data = value;  temp->next = 0;  return temp;  }  // Erase member function  template <typename T>  void List <T> ::erase(int pos)  {  if (pos < 0 || pos >(count - 1))  {  cout << "Error! The position is out of range." << endl;  return;  }  if (pos == 0)  {  Node <T>\* del = begin;  begin = begin->next;  delete del;  }  else  {  Node <T>\* cur = begin;  for (int i = 0; i < pos - 1; i++)  {  cur = cur->next;  }  Node <T>\* del = cur->next;  cur->next = cur->next->next;  delete del;  }  count--;  }  // Get member function  template <typename T>  T& List <T> ::get(int pos) const  {  if (pos < 0 || pos > count - 1)  {  cout << "Error! Position out of range.";  assert(false);  }  else if (pos == 0)  {  return begin->data;  }  else  {  Node <T>\* cur = begin;  for (int i = 0; i < pos; i++)  {  cur = cur->next;  }  return cur->data;  }  }  // Size member function  template <typename T>  int List <T> ::size() const  {  return count;  }  // Implement print member function  // Print member function  template <typename T>  void List <T> ::print() const  {  if (count == 0)  {  cout << "List is empty!" << endl;  return;  }  Node <T>\* cur = begin;  while (cur != 0)  {  cout << cur->data << endl;  cur = cur->next;  }  }  // Implement reversePrint member function  template <typename T>  void List <T> ::reversePrint() const  {  if (count == 0)  {  cout << "List is empty!" << endl;  return;  }  Node <T>\* cur = begin;  RPHelper(begin);  }  template <typename T>  void List <T> ::RPHelper(Node<T>\* head) const {  if (head == 0) {  return;  }  RPHelper(head->next);  cout << head->data << endl;  } |
| Screenshot |
| 텍스트, 스크린샷이(가) 표시된 사진  자동 생성된 설명 |

|  |
| --- |
| Q3 |
| Code |
| #include <iostream>  #include <vector>  #include <algorithm>  #include <string>  #include <functional>  using namespace std;  void print1(string value) {  cout << value << " ";  }  void print2(int value) {  cout << value << " ";  }  int main() {  std::vector<string> names;  names.push\_back("John");  names.push\_back("Mary");  names.push\_back("Lucie");  names.push\_back("Robert");  names.push\_back("Suzan");  names.push\_back("Richard");  cout << "Unsorted Names: " << endl;  for\_each(names.begin(), names.end(), print1);  cout << endl << endl;  sort(names.begin(), names.end());  cout << "Sorted Names: " << endl;  for\_each(names.begin(), names.end(), print1);  cout << endl << endl;  std::vector<int> integers;  integers.push\_back(17);  integers.push\_back(23);  integers.push\_back(11);  integers.push\_back(18);  cout << "Unsorted Integers: " << endl;  for\_each(integers.begin(), integers.end(), print2);  cout << endl << endl;  sort(integers.begin(), integers.end());  cout << "Sorted Integers: " << endl;  for\_each(integers.begin(), integers.end(), print2);  cout << endl << endl;  printf("\n Press Any Key to Terminate ...\n");  getchar();  return 0;  } |
| Screenshot |
| 텍스트, 스크린샷, 폰트이(가) 표시된 사진  자동 생성된 설명 |