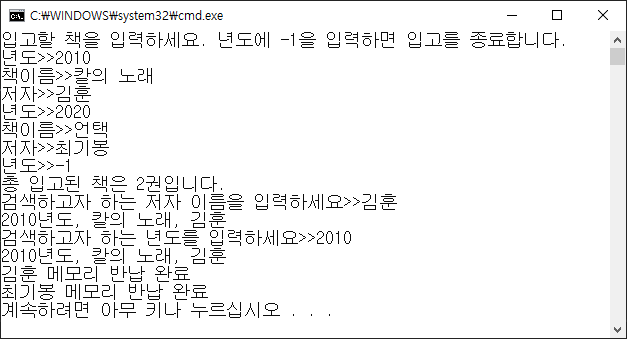
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2020\_2\_C++ \_14 | 학번 : | 20175105 | 이름 : | 곽영주 |

* **프로그램 과제**

1. 벡터를 사용하여 책 정보를 저장하고 저자와 년도로 검색하는 프로그램을 완성하시오

class Book {

 string title; // 책 이름

string author; // 저자

int year; //출판년도

public:

Book() {};

Book(string title, string author, int year) {

this->title = title;

this->author = author;

this->year = year;

}

string getAuthor() { return author; }

int getYear() { return year; }

void show() {

cout << year << "년도, " << title << ", " << author << endl;

}

};

class BookManager {

vector<Book\*> v; //Book 객체를 저장하기위한 vetor 객체 생성

void searchByAuthor(); //저자로 검색

void searchByYear(); //연도로 검색

void bookIn(); //벡터에 Book 정보 저장

void finish();

public:

void run();

};

void BookManager::run() {

bookIn(); // 입고

searchByAuthor(); // 저자로 검색

searchByYear(); // 년도로 검색

finish();

}

int main() {

BookManager man;

man.run();

}

|  |
| --- |
| [프로그램 소스]  #include <iostream>  #include <string>  #include <vector>  using namespace std;  class Book {  string title;  string author;  int year;  public:  //Book() {};  Book(string title = "", string author = "", int year = 0) {  this->title = title;  this->author = author;  this->year = year;  }  string getAuthor() { return author; }  int getYear() { return year; }  void show() { cout << year << "년도, " << title << ", " << author << endl; }  };  class BookManager {  vector<Book\*> v;  void searchByAuthor();  void searchByYear();  void bookIn();  void finish();  public:  void run();  };  void BookManager::searchByAuthor() {  string author;  cout << "검색하고자 하는 저자 이름을 입력하세요>> ";  getline(cin, author, '\n');  for (int i = 0; i < v.size(); i++) {  if (v[i]->getAuthor() == author) {  v[i]->show();  return;  }  }  cout << "검색한 저자로 된 책이 없습니다." << endl;  }  void BookManager::searchByYear() {  int year;  cout << "검색하고자 하는 년도를 입력하세요>> ";  cin >> year;  for (int i = 0; i < v.size(); i++) {  if (v[i]->getYear() == year) {  v[i]->show();  return;  }  }  cout << "검색한 년도로 된 책이 없습니다." << endl;  }  void BookManager::bookIn() {  string title, author;  int year;  cout << "입고할 책을 입력하세요. 년도에 -1을 입력하면 입고를 종료합니다." << endl;  while (true) {  cout << "년도>> ";  cin >> year;  cin.ignore();  if (year == -1) {  cout << "총 입고된 책은 " << v.size() << "권 입니다." << endl;  return;  }  cout << "책이름>> ";  getline(cin, title, '\n');  cout << "저자>> ";  getline(cin, author, '\n');  v.push\_back(new Book(title, author, year));  }  }  void BookManager::finish() {  for (int i = 0; i < v.size(); i++) {  cout << v[i]->getAuthor() + " 메모리 반납 완료" << endl;  delete v[i];  }  }  void BookManager::run() {  bookIn();  searchByAuthor();  searchByYear();  finish();  }  int main() {  BookManager man;  man.run();  return 0;  } |
| [실행 결과] |

1. 배열에서 원소를 검색하는 search() 함수를 템플릿으로 작성하시오. search()의 첫번째 매개변수는 검색하고자 하는 원소 값이고 두번째 매개변수는 배열이며, 세번째 매개변수는 배열의 개수이다. 검색에 성공하면 true, 실패하면 false를 반환한다. 호출 사례는 다음과 같다. double, char 타입의 배열도 적용하여 결과를 제시하시오

int x[] = { 1, 10, 100, 5, 4 };

if (search(100, x, 5))

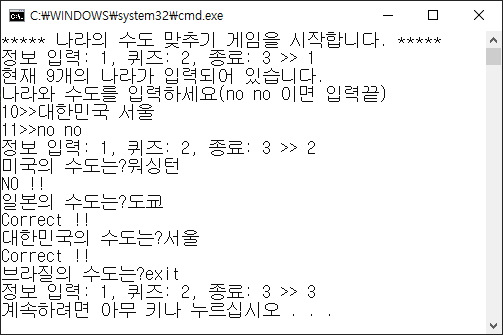
cout << "100이 배열 x에 포함되어 있다";

else

cout << "100이 배열 x에 포함되어 있지 않다";

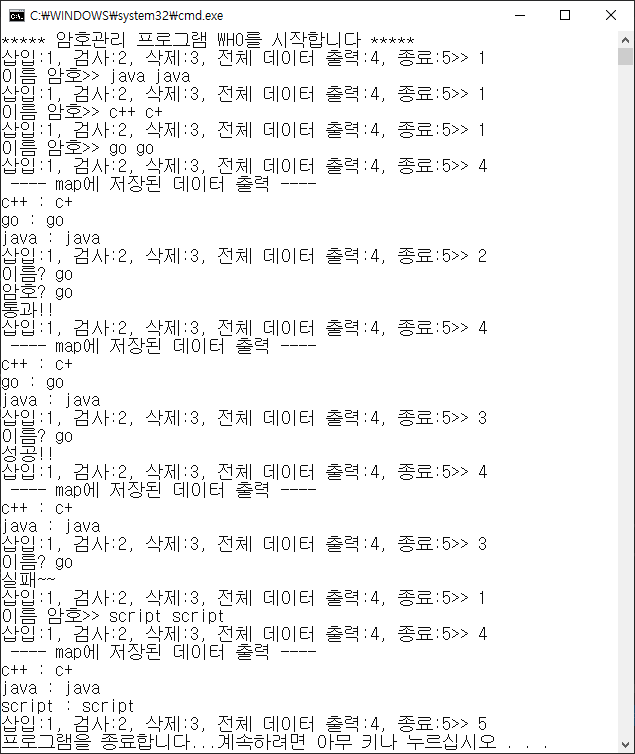
|  |
| --- |
| [프로그램 소스]  #include <iostream>  using namespace std;  template <class T> bool search(T n, T\* arr, int size) {  for (int i = 0; i < size; i++) {  if (arr[i] == n)  return true;  }  return false;  }  int main() {  int x[] = { 1,10,100,5,4 };  double d[] = { 1.1,2.3,12.5 };  char ch[] = { 'a','B','c','D' };  if (search(100, x, 5))  cout << "100이 배열 x에 포함되어 있다." << endl;  else  cout << "100이 배열 x에 포함되어 있지 않다." << endl;  if (search(15.3, d, 3))  cout << "15.3가 배열 d에 포함되어 있다." << endl;  else  cout << "15.3이 배열 d에 포함되어 있지 않다." << endl;  if (search('a', ch, 4))  cout << "a가 배열 ch에 포함되어 있다." << endl;  else  cout << "a가 배열 ch에 포함되어 있지 않다." << endl;  return 0;  } |
| [실행 결과] |

1. 나라의 수도 맞추기 게임을 vector를 사용하여 작성하시오. 나라 이름과 수도 문자열로 구성된 Nation클래스를 만들고 vector<Nation> v;로 생성한 벡터를 이용하여 나라 이름과 수도 이름을 삽입할 수도 있고 랜덤하게 퀴즈를 볼 수도 있다. 프로그램 내에서 벡터에 Nation객체를 여러 개 미리 삽입하여 처리한다.



|  |
| --- |
| [프로그램 소스]  #include <iostream>  #include <string>  #include <vector>  #include <cstdlib>  #include <ctime>  using namespace std;  class Nation {  string name; // 나라 이름  string capital; // 수도  public:  Nation(string name = "", string capital = "") {  this->name = name;  this->capital = capital;  }  string getName() { return name; }  string getCapital() { return capital; }  };  class NationGame {  vector<Nation> v;  void data(); // 미리 삽입된 퀴즈 데이터  void add(); // 정보 입력  void quiz(); // 퀴즈  public:  void run();  };  void NationGame::data() {  v.push\_back(Nation("대한민국", "서울"));  v.push\_back(Nation("미국", "워싱턴D.C."));  v.push\_back(Nation("일본", "도쿄"));  v.push\_back(Nation("중국", "베이징"));  }  void NationGame::add() {  string name, capital;  int index = v.size() + 1;  cout << "현재 " << v.size() << "개의 나라가 입력되어 있습니다." << endl;  cout << "나라와 수도를 입력하세요. (no no 이면 입력 끝)" << endl;  while (true) {  cout << index << ">> ";  cin >> name >> capital;  if (name == "no" && capital == "no")  return;  v.push\_back(Nation(name, capital));  index++;  }  }  void NationGame::quiz() {  string capital;  int index;  // 랜덤으로 생성되는 퀴즈 문제의 중복을 없애기 위해  vector<Nation> temp;  vector<Nation>::iterator it;  temp = v;    while (!temp.empty()) {  it = temp.begin();  index = rand() % temp.size();  cout << temp[index].getName() << "의 수도는? ";  cin >> capital;  if (capital == "exit")  break;  if (capital == temp[index].getCapital())  cout << "Correct !!" << endl;  else  cout << "NO !!" << endl;  temp.erase(it + index);  }  }  void NationGame::run() {  int num;  srand((unsigned)time(0));  cout << "\*\*\*\*\* 나라의 수도 맞추기 게임을 시작합니다. \*\*\*\*\*" << endl;  data();  while (true) {  cout << "정보 입력: 1, 퀴즈: 2, 종료: 3 >> ";  cin >> num;  switch (num) {  case 1:  add();  break;  case 2:  quiz();  break;  default:  return;  }  }  }  int main() {  NationGame game;  game.run();  return 0;  } |
| [실행 결과] |

1. 암호 관리 응용 프로그램을 map을 이용하여 작성하시오.



|  |
| --- |
| [프로그램 소스]  #include <iostream>  #include <string>  #include <map>  using namespace std;  class MapManager {  map<string, string> m;  map<string, string>::iterator it;  void add(); // 삽입  void check(); // 검사  void del(); // 삭제  void show(); // 출력  public:  void run();  };  void MapManager::add() {  string name, password;  cout << "이름 암호>> ";  cin >> name >> password;  m.insert(make\_pair(name, password));  }  void MapManager::check() {  string name, password;  cout << "이름? ";  cin >> name;  cout << "암호? ";  cin >> password;  it = m.find(name);  if (it == m.end())  cout << "존재하지 않는 이름 입니다." << endl;  else {  if (m.at(name) == password)  cout << "통과!!" << endl;  else  cout << "암호가 틀렸습니다." << endl;  }  }  void MapManager::del() {  string name;  cout << "이름? ";  cin >> name;  it = m.find(name);  if (it == m.end())  cout << "실패~~" << endl;  else {  m.erase(it); //  cout << "성공!!" << endl;  }  }  void MapManager::show() {  cout << " ---- map에 저장된 데이터 출력 ---- " << endl;  for (it = m.begin(); it != m.end(); ++it)  cout << it->first << " : " << it->second << endl;  }  void MapManager::run() {  int num;  cout << "\*\*\*\*\* 암호관리 프로그램 WHO를 시작합니다. \*\*\*\*\*" << endl;  while (true) {  cout << "삽입:1, 검사:2, 삭제:3, 전체 데이터 출력:4, 종료:5>> ";  cin >> num;  switch (num) {  case 1:  add();  break;  case 2:  check();  break;  case 3:  del();  break;  case 4:  show();  break;  default:  return;  }  }  }  int main() {  MapManager mm;  mm.run();  return 0;  } |
| [실행 결과] |