|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2019\_1\_C++ \_15 | 학번 : | 20145165 | 이름 : | 정균모 |

* getline() 으로 파일 읽고 단어 검색
* 메모장에서 “words.txt” 파일을 작성하여 영단어를 저장한다
* 프로그램과 동일한 폴더에 있어야 함

|  |
| --- |
| #include <iostream>  #include <fstream>  #include <string>  #include <vector>  using namespace std;  void fileRead(vector<string> &v, ifstream &fin) { // fin 스트림으로부터 벡터 v에 읽어들임  string line;  while(getline(fin, line)) { // fin 파일에서 한 라인 읽기  v.push\_back(line); // 읽은 라인을 벡터에 저장  }  }  void search(vector<string> &v, string word) { // 벡터 v에서 word를 찾아 출력  for(int i=0; i<v.size(); i++) {  int index = v[i].find(word);  if(index != -1) // found  cout << v[i] << endl;  }  }  int main() {  vector<string> wordVector;  ifstream fin("words.txt");  if(!fin) {  cout << "words.txt 파일을 열 수 없습니다" << endl;  return 0; // 열기 오류  }  fileRead(wordVector, fin); // fin 스트림으로부터 wordVector에 라인 별로 읽기  fin.close();  cout << "words.txt 파일을 읽었습니다." << endl;  while(true) {  cout << "검색할 단어를 입력하세요 >>";  string word;  getline(cin, word); // 키보드로부터 문자열 읽기  if(word == "exit")  break; // 프로그램 종료  search(wordVector, word); // 문자열을 words.txt에서 검색하여 출력  }  cout << "프로그램을 종료합니다." << endl;  } |
| **[실행 결과]** |

* 바이너리 I/O - 블록단위 입출력

|  |
| --- |
| #include <iostream>  #include <fstream>  using namespace std;  int main() {  const char\* file = "c:\\temp\\data.dat";  ofstream fout;  fout.open(file, ios::out | ios::binary); // 읽기 모드로 파일 열기  if(!fout) { // 열기 실패 검사  cout << "파일 열기 오류";  return 0;  }  int n[] = {0,1,2,3,4,5,6,7,8,9};  double d = 3.15;  fout.write((char\*)n, sizeof(n)); // int 배열 n을 한 번에 파일에 쓴다.  fout.write((char\*)(&d), sizeof(d)); // double 값 하나를 파일에 쓴다.  fout.close();  // 배열 n과 d 값을 임의의 값으로 변경시킨다.  for(int i=0; i<10; i++) n[i]=99;  d = 8.15;  // 배열 n과 d 값을 파일에서 읽어 온다.  ifstream fin(file, ios::in);  if(!fin) { // 열기 실패 검사  cout << "파일 열기 오류";  return 0;  }  fin.read((char\*)n, sizeof(n)); // int 배열을 한 번에 읽오 온다.  fin.read((char\*)(&d), sizeof(d)); // double 값을 읽어 온다.  for(int i=0; i<10; i++) // 읽은 배열 n을 확인한다.  cout << n[i] << ' ';  cout << endl << d << endl; // 읽은 double 값을 확인한다.  fin.close();  } |
| **[실행 결과]** |

* 프로그램 과제

1. 교재 639p : 문제 7번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  #include <fstream>  #include <string>  #include <vector>  using namespace std;  void fileRead(vector<string> &v, ifstream &fin) {  string line;  while (getline(fin, line)) {  v.push\_back(line);  }  }  int main() {  const char\* fileReadini = "c:\\Users\\User\\Documents\\C++\\system.ini";  vector<string> wordvector;  ifstream fin(fileReadini, ios::binary);  if (!fin) { // 열기 실패 검사  cout << "파일 열기 오류";  return 0;  }  fileRead(wordvector, fin);    const char\* fileWritetxt = "c:\\Users\\User\\Documents\\C++\\system.txt";  ofstream fout;  fout.open(fileWritetxt, ios::out | ios::binary);  if (!fout) { // 열기 실패 검사  cout << "파일 열기 오류";  return 0;  }  reverse(wordvector.begin(), wordvector.end());  for (int i = 0; i < wordvector.size(); i++) {  fout << wordvector.at(i)<<endl;  }  fin.close();  fout.close();  system("pause");  } |
| **[실행 결과]** |

1. 교재 641p : 문제 13번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  #include <fstream>  #include <string>  #include <vector>  using namespace std;  void fileRead(vector<string> &v, ifstream &fin) { // fin 스트림으로부터 벡터 v에 읽어들임  string line;  while (getline(fin, line)) { // fin 파일에서 한 라인 읽기  v.push\_back(line); // 읽은 라인을 벡터에 저장  }  }  int main() {  vector<string> wordVector;  string Item;  bool sw = false;  ifstream fin("words.txt");  if (!fin) {  cout << "words.txt 파일을 열 수 없습니다" << endl;  return 0; // 열기 오류  }  fileRead(wordVector, fin); // fin 스트림으로부터 wordVector에 라인 별로 읽기  fin.close();  cout << "words.txt 파일 로딩 완료" << endl <<"검색을 시작합니다. 단어를 입력해 주세요."<< endl;  while (true) {  cout << "단어 >>";  string word;  getline(cin, word); // 키보드로부터 문자열 읽기  if (word == "exit")  break; // 프로그램 종료  for (int i = 0; i < wordVector.size(); i++) {  Item = wordVector.at(i);  if (Item.find(word) == 0) {  cout << Item << endl;  sw = true;  }  }  if (sw==false) {  cout << "발견하지 못했습니다."<<endl;  }  sw = false;  }  cout << "프로그램을 종료합니다." << endl;  } |
| **[실행 결과]** |