**REPORT**

심화프로그래밍 실습

6주차

**과목명   |** 심화프로그래밍

**담당교수 |** 김가영 교수님

**학과     |** 컴퓨터공학과

**학번     |** 2018112028

**이름     |** 김지훈

**제출일   |**  2020.10.16

실습 1번)

|  |
| --- |
| 문제 분석 :  n개의 숫자(size 지정 x)와 연산자를 입력 받아 가능한 모든 수식을 생성한다. 단, 숫자의 순서는 바꾸면 안된다.  생성된 수식 중에서 제일 큰 수와 작은 수를 재귀적으로 구한다.  저번 실습 class에 getSize() , setStringNum(?), getNum(), changeNum()만 추가한다.  아래의 코드를 실행 했을 때 A의 값이 변화하지 않는다. 1) 변화하지 않는 이유를 작성하시오. (교안 스택 그림 참고)  2)목적대로 수행할 수 있도록 void change(cal A, cal B) 함수를 수정하고 3)수정한 이유를 작성 하시오. |
| 프로그램 설계 / 알고리즘 : |
| 소스코드 / 주석 : |
| 결과 / 결과 분석 :       * 두 결과 모두 max와 min이 잘 계산된다. |

실습 2번)

|  |
| --- |
| 문제 분석 :  Person 클래스와 main함수를 작성하여, 3개의 Person 객체를 가지는 배열을 선언하고, 다음과 같이 이름과 전화번호를 입력 받아 출력하고 검색하는 프로그램을 완성하라 |
| 프로그램 설계 / 알고리즘 :  주어진 클래스를 작성하고, 사람 3명에 맞게 main을 작성한다. 클래스 중 메소드 내용을 추가 작성한다. |
| 소스코드 / 주석 :   |  | | --- | | //2018112028 김지훈  #include <iostream>  #include <string>  using namespace std;  class Person  {  private:  string name;  string tel;    public:  string getName() { return name; }  string getTel() { return tel; }  void set(string name, string tel)  {  this->name = name;  this->tel = tel;  }  };  int main()  {  // 이름, 전화번호 입력  cout << "이름과 전화 번호를 입력해 주세요" << endl;  Person person[3];  string n, t;  for (int i = 0; i < 3; ++i)  {  cout << "사람 " << i + 1 << ">> ";  cin >> n >> t;  person[i].set(n, t);  }  // 출력  cout << "모든 사람의 이름은 ";  for (int i = 0; i < 3; ++i)  cout << person[i].getName() << ' ';  cout << endl << "전화번호 검색합니다. 이름을 입력하세요>>";  cin >> n;  for (int i = 0; i < 3; ++i)  {  if (n == person[i].getName())  cout << "전화번호는 " << person[i].getTel() << endl;  }  } | |
| 결과 / 결과 분석 :     * 사람의 이름과 전화번호가 잘 저장되며, 출력된다. |

실습 3-2번)

|  |
| --- |
| 문제 분석 :  Person 클래스,ManagePeople 클래스, main함수를 작성한다. cpp, h 파일 분리한다.  ManagePeople 객체를 활용해 Person 객체를 관리한다.  setMember()에서 Person 정보를 입력 받고 Show() 에서 모든 Person의 이름을 출력한다. Search()함수에서는 이름을 통해 전화번호를 검색한다.  모든 객체는 객체 포인터를 사용한다. |
| 프로그램 설계 / 알고리즘 :  2번의 main.cpp에 쓰이던 기능을 ManagePeople클래스로 이전한다. 또한 객체의 개수를 입력하고 입력한 수만큼 member를 동적 할당 해준다. |
| 소스코드 / 주석 :  main.cpp   |  | | --- | | //2018112028 김지훈  #include <iostream>  #include "ManagePeople.h"  using namespace std;  int main()  {  ManagePeople m1;  m1.setMember();  m1.show();  m1.search();  } |   ManagePeople.h   |  | | --- | | //2018112028 김지훈  #pragma once  #include <iostream>  #include "Person.h"  using namespace std;  class ManagePeople  {  private:  Person \*member; // Person 객체  int num; // Person 객체의 수  public:  ManagePeople();  ~ManagePeople();  void setMember(); // Person 정보 입력  void show(); // 모든 Person의 이름을 출력  void search(); // 이름을 통해 전화번호를 검색한다.  }; |   ManagePeople.cpp   |  | | --- | | //2018112028 김지훈  #include <iostream>  #include <string>  #include "ManagePeople.h"  #include "Person.h"  using namespace std;  ManagePeople::ManagePeople()  {  cout << "생성할 Person 객체의 개수를 입력하시오: ";  cin >> this->num;  this->member = new Person[this->num];  }  ManagePeople::~ManagePeople()  {  }  void ManagePeople::setMember() // Person 정보 입력  {  cout << "이름과 전화 번호를 입력해 주세요" << endl;  string n, t;  for (int i = 0; i < this->num; ++i)  {  cout << "사람 " << i + 1 << ">> ";  cin >> n >> t;  member[i].set(n, t);  }  }  void ManagePeople::show() // 모든 Person의 이름을 출력  {  cout << "모든 사람의 이름은 ";  for (int i = 0; i < this->num; ++i)  cout << member[i].getName() << ' ';  }  void ManagePeople::search() // 이름을 통해 전화번호를 검색한다.  {  string n;  cout << endl << "전화번호 검색합니다. 이름을 입력하세요>>";  cin >> n;  for (int i = 0; i < 3; ++i)  {  if (n == member[i].getName())  cout << "전화번호는 " << member[i].getTel() << endl;  }  } |   Person.h   |  | | --- | | //2018112028 김지훈  #pragma once  #include <iostream>  #include <string>  using namespace std;  class Person  {  private:  string name;  string tel;  public:  string getName();  string getTel();  void set(string name, string tel);  }; |   Person.cpp   |  | | --- | | //2018112028 김지훈  #include <iostream>  #include <string>  #include "Person.h"  using namespace std;  string Person::getName() { return name; }  string Person::getTel() { return tel; }  void Person::set(string name, string tel)  {  this->name = name;  this->tel = tel;  } | |
| 결과 / 결과 분석 :     * 객체가 잘 생성되고, 잘 출력된다. |

실습 4번)

|  |
| --- |
| 문제 분석 :  두사람이 게임을 진행하며 선수의 이름을 초기에 입력 받는다.  선수가 번갈아 자신의 차례에서 엔터키를 치면 랜덤한 3개의 수가 생성되고 모두 동일한 수 가 나오면 게임에서 이기게 됨  숫자의 범위가 너무 크면 3개의 숫자가 일치할 가능성이 낮아 숫자의 범위를 0~2로 제한  선수는 Player클래스로 작성하고 2명의 선수는 배열로 구성 |
| 프로그램 설계 / 알고리즘 :  클래스를  class Player  {  private:  string name;  public:  void setName(string name); // 이름 설정  string getName(); // 이름 리턴  bool play(); // 게임 시작 & 게임 승리여부 판단  };  위와 같이 만들어서 Player의 이름 설정과 게임플레이를 한번에 하도록한다. |
| 소스코드 / 주석 :   |  | | --- | | //2018112028 김지훈  #include <iostream>  #include <string>  #include <random>  using namespace std;  class Player  {  private:  string name;  public:  void setName(string name); // 이름 설정  string getName(); // 이름 리턴  bool play(); // 게임 시작 & 게임 승리여부 판단  };  void Player::setName(string name)  {  this->name = name;  }  string Player::getName()  {  return this->name;  }  bool Player::play()  {  int num[3];  string tmp;  random\_device rng;  cout << this->getName() << " : <ENTER>";  getline(cin, tmp);  for (int i = 0; i < 3; i++) // 랜덤한 숫자 배정  {  num[i] = rng() % 3; // 0~2  cout << num[i] << " ";  }    if (num[0] == num[1] && num[1] == num[2]) // 승리 시  {  cout << this->getName() << "님 승리!" << endl;  return true;  }  else // 패배 시  {  cout << "아쉽군요!" << endl;  return false;  }  }  int main()  {  Player \*p = new Player[2]; // Player 변수 선언  string name;  cout << "첫 번째 선수 이름 >> "; // 선수 이름 설정  getline(cin, name);  p[0].setName(name);  cout << "두 번째 선수 이름 >> ";  getline(cin, name);  p[1].setName(name);  int turn = 1; // 플레이 하는 선수 턴  do // 턴을 번갈아 가면서 play  {  if (turn == 0)  turn = 1;  else if (turn == 1)  turn = 0;  } while (!p[turn].play());  } | |
| 결과 / 결과 분석 :     * 난수도 매번 무작위로 잘 생성되어 게임 플레이가 잘된다. |

소감 : 클래스가 여러 개일 경우, 헤더와 cpp파일을 나누어 코딩하는 것이 관리하는 면에서 많이 편했다. 앞으로 번거롭더라도 나중을 위해 이렇게 작업하는 습관을 가져야겠다.