**객체지향프로그래밍 LAB #10**

**<기초문제>\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. 아래의 프로그램을 작성하시오. (/\*구현\*/ 부분을 채울 것, 표의 상단: 소스코드, 하단: 실행결과)

|  |
| --- |
| #include <iostream>  #include <string>  using namespace std;  class Point {  private: // class 안에서만 사용가능  int x; // 멤버 변수  int y;  public: // class 안/밖에서 사용가능  // Point() {}  /\*구현\*/ { //constructor: class와 이름이 같다,  x = \_x;  y = \_y;  }  /\*X,Y set함수 구현(함수 1개)\*/  /\*X,Y get 함수 각각 구현(함수 2개)\*/  void print() {  cout << x << ", " << y << endl;  }  };  int main() {  Point pt1(1, 2), pt2(3, 4);  pt1.setXY(10, 20);  pt1.print();  pt2.print();  cout << pt1.getX() << endl;  cout << pt2.getY() << endl;  return 0;  } |
|  |
| #include <iostream>  #include <string>  using namespace std;  class Point {  private: // class 안에서만 사용가능  int x; // 멤버 변수  int y;  public: // class 안/밖에서 사용가능  //Point() {}  Point(int \_x, int \_y) {  x = \_x;  y = \_y;//constructor: class와 이름이 같다,  }  /\*X,Y set함수 구현(함수 1개)\*/  void setXY(int \_x, int \_y) {  x = \_x;  y = \_y;  }  /\*X,Y get 함수 각각 구현(함수 2개)\*/  int getX() {  return x;  }  int getY() {  return y;  }  void print() {  cout << x << ", " << y << endl;  }  };  int main() {  Point pt1(1, 2), pt2(3, 4);  pt1.setXY(10, 20);  pt1.print();  pt2.print();  cout << pt1.getX() << endl;  cout << pt2.getY() << endl;  return 0;  } |
|  |

2. 아래의 프로그램을 작성하시오. (/\*구현\*/ 부분을 채울 것)

|  |
| --- |
| #include <string>  #include <iostream>  using namespace std;  class Account {  //private: //멤버변수  string name;  string id;  double balance;  public: // 멤버함수(method)  // 3개의 생성자 구현  /\*구현\*/ // 기본생성자. name : "", id : "", balance : 0 으로 초기화  /\*구현\*/ // name, id 받아오고, balance는 0으로 초기화  /\*구현\*/ // name, id , balance 받아와서 초기화, balance < 0 인 경우 0으로 초기화  void deposit(double \_amt) { balance += \_amt; }  bool withdraw(double \_amt) {  if (balance - \_amt < 0)  return false;  balance -= \_amt;  return true;  }  void print() {  cout << name << ", " << id << ", " << balance << endl;  }  };  int main() {  Account ac1("배성호", "1002", 5000);  Account ac2;  ac2.print();  ac1.print();  int depo;  cout << "예금할 금액을 입력하세요 : ";  cin >> depo;  ac1.deposit(depo);  /\*구현\*/ // print() 함수로 ac1 출력  int wdrw;  cout << "출금할 금액을 입력하세요 : ";  cin >> wdrw;  if (!ac1.withdraw(wdrw))  cout << "잔액이 부족합니다." << endl;  ac1.print();  return 0;  } |
|  |
| #include <string>  #include <iostream>  using namespace std;  class Account {  private: //멤버변수: default는 prvate  string name;  string id;  double balance;  public: // 멤버함수(method)  // 3개의 생성자 구현  Account() : name(""), id(""), balance(0) {} // 기본생성자. name : "", id : "", balance : 0 으로 초기화  Account(string \_name, string \_id) : name(\_name), id(\_id), balance(0) {}/\*구현\*/ // name, id 받아오고, balance는 0으로 초기화  Account(string \_name, string \_id, double \_balance) {  name = \_name;  id = \_id;  if (\_balance < 0) {  cout << "Error, negative balance" << endl;  balance = 0;  }  else balance = \_balance;  } // name, id , balance 받아와서 초기화, balance < 0 인 경우 0으로 초기화  void deposit(double \_amt) { balance += \_amt; }  bool withdraw(double \_amt) {  if (balance - \_amt < 0)  return false;  balance -= \_amt;  return true;  }  void print() {  cout << name << ", " << id << ", " << balance << endl << endl;  }  };  int main() {  Account ac1("배성호", "1002", 5000);  Account ac2;  ac2.print();  ac1.print();  int depo;  cout << "예금할 금액을 입력하세요 : ";  cin >> depo;  ac1.deposit(depo);  ac1.print(); // print() 함수로 ac1 출력  int wdrw;  cout << "출금할 금액을 입력하세요 : ";  cin >> wdrw;  if (!ac1.withdraw(wdrw))  cout << "잔액이 부족합니다." << endl;  ac1.print();  return 0;  } |
|  |

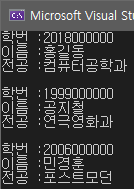
**<응용문제>\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. 주어진 main 함수는 학번, 이름, 전공을 입력받아 출력한다. 주어진 코드를 기반으로 아래의 조건에 맞게 CStudent 클래스를 작성하시오.

1. 기본 생성자는 initialization list를 이용하여 초기화함.
2. 학번, 이름, 전공을 파라미터로 받아 초기화하는 생성자도 작성함.
3. 학번, 이름, 전공 각각에 대해 파라미터로 받아서 설정하는 멤버함수 및 클래스 외부로 해당 멤버 변수들을 가져올 수 있도록 하는 멤버함수 설정.
4. 멤버함수 Display는 출력화면과 같이 화면에 출력해야 함.
5. 학번(int), 이름(string), 전공(string) 각각 멤버변수가 존재하고 private으로 설정.

|  |
| --- |
| int main() {  CStudent s1; // A  s1.Display();  CStudent s2(1999000000, "공지철", "연극영화과"); // B  s2.Display();  // C  s1.setNumber(2006000000);  s1.setName("민경훈");  s1.setMajor("포스트모던");  cout << "학번 :" << s1.getNumber() << endl;  cout << "이름 :" << s1.getName() << endl;  cout << "전공 :" << s1.getMajor() << endl;  return 0;  } |

1-출력화면:

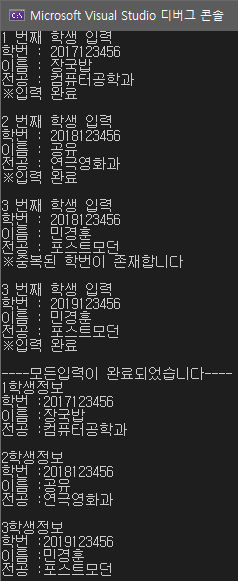


|  |
| --- |
| #include <string>  #include <iostream>  using namespace std;  class CStudent {  private:  int m\_Number;  string m\_Name;  string m\_Major;  public:  CStudent() : m\_Number(2018000000), m\_Name("홍길동"), m\_Major("컴퓨터공학과") {}  CStudent(int number, string name, string major) : m\_Number(number), m\_Name(name), m\_Major(major) {}  void setNumber(int number) {  m\_Number = number;  }  void setName(string name) {  m\_Name = name;  }  void setMajor(string major) {  m\_Major = major;  }  int getNumber() {  return m\_Number;  }  string getName() {  return m\_Name;  }  string getMajor() {  return m\_Major;  }  void Display() {  cout << "학번 : " << getNumber() << endl;  cout << "이름 : " << getName() << endl;  cout << "전공 : " << getMajor() << endl << endl;  }  };  int main() {  CStudent s1; // A  s1.Display();  CStudent s2(1999000000, "공지철", "연극영화과"); // B  s2.Display();  // C  s1.setNumber(2006000000);  s1.setName("민경훈");  s1.setMajor("포스트모던");  cout << "학번 :" << s1.getNumber() << endl;  cout << "이름 :" << s1.getName() << endl;  cout << "전공 :" << s1.getMajor() << endl;  return 0;  } |
|  |

2. 1번 문제에서 작성한 CStudent 클래스를 사용하여 최대 3명의 학생 정보를 입력받고 출력하는 프로그램을 작성하시오. 단, 학번은 고유하기 때문에 Primary key로 설정하고 중복되지 않도록 구현함.

|  |
| --- |
| int main() {  CStudent s[3]; // 3명의 학생정보를 저장할 class 배열  int inputNumber; // 키보드로 학번을 입력 받을 변수  string inputName, inputMajor; // 키보드로 이름, 전공을 입력 받을 변수  int length = 0; // 현재 입력된 학생의 수  while (1) { /\* 구현 \*/ }  return 0;  } |

2-출력화면:



|  |
| --- |
| #include <string>  #include <iostream>  using namespace std;  class CStudent {  private:  int m\_Number;  string m\_Name;  string m\_Major;  public:  CStudent() : m\_Number(2018000000), m\_Name("홍길동"), m\_Major("컴퓨터공학과") {}  CStudent(int number, string name, string major) : m\_Number(number), m\_Name(name), m\_Major(major) {}  void setNumber(int number) {  m\_Number = number;  }  void setName(string name) {  m\_Name = name;  }  void setMajor(string major) {  m\_Major = major;  }  int getNumber() {  return m\_Number;  }  string getName() {  return m\_Name;  }  string getMajor() {  return m\_Major;  }  void Display() {  cout << "학번 : " << getNumber() << endl;  cout << "이름 : " << getName() << endl;  cout << "전공 : " << getMajor() << endl<<endl;  }  };  int main() {  CStudent s[3]; // 3명의 학생정보를 저장할 class 배열  int inputNumber; // 키보드로 학번을 입력 받을 변수  string inputName, inputMajor; // 키보드로 이름, 전공을 입력 받을 변수  int length = 0; // 현재 입력된 학생의 수  while (1) {  int d = 0;  if (length < 3) {  cout << length + 1 << " 번째 학생 입력" << endl;  cout << "학번 : ";  cin >> inputNumber;  cout << "이름 : ";  cin >> inputName;  cout << "전공 : ";  cin >> inputMajor;  if (length == 0) {  s[0].setNumber(inputNumber);  s[0].setName(inputName);  s[0].setMajor(inputMajor);  cout << "※입력 완료" << endl << endl;  d++;  }  else {  for (int i = 0; i < length; i++) {  if (s[i].getNumber() == inputNumber) {  cout << "※중복된 학번이 존재합니다" << endl << endl;  length--;  d++;  break;  }  else if (i == length - 1) {  s[length].setName(inputName);  s[length].setNumber(inputNumber);  s[length].setMajor(inputMajor);    }    }    }    }  else {  cout << "----모든 입력이 완료되었습니다----" << endl;  for (int i = 0; i < length; i++) {  cout << i + 1 << "학생정보" << endl;  s[i].Display();  }  break;  }  length++;  if (d == 0) cout << "※입력 완료" << endl << endl;  }  return 0;  } |
|  |

3. 다음에 따라 학사관리 프로그램을 작성하시오.

1) Student 클래스 멤버변수

* 이름(string), 학번(int), 학과(string)를 멤버변수로 생성함.
* 학생이 들은 과목(vector<string>)과 성적(vector<char>)을 저장하는 멤버변수를 생성함.
* 모든 멤버변수는 private으로 설정함.

2) Student 클래스의 멤버함수들은 모두 public으로 생성하시오. 아래와 같이 두 가지 방식으로 main 함수에서 Student 객체를 생성할 수 있도록 생성자를 만드시오.

* Student Harry(“Harry”, 2018101234, “SWCON”);
* Student Ron;  
  // 이때 Initialization list를 이용하여 초기값(“default”, 0, “depart”)을 설정하시오.

3) Student Ron과 같은 방식으로 객체를 생성한 경우, 추후 이름, 학번, 학과를 세팅할 수 있도록 setName, setID, setDpt 함수를 각각 만드시오. Student 객체의 정보를 다음과 같이 한 줄에 출력할 수 있도록 print 함수를 만드시오.

개체이(가) 표시된 사진

자동 생성된 설명

4) 학생의 성적을 입력하는 addGrade 함수를 구현하시오. 이 함수는 과목(string) 하나와 해당 과목의 성적(char)을 파라미터로 받아 과목 이름은 과목 vector 가장 마지막에, 성적은 성적 vector 가장 마지막에 저장한다. 과목과 성적 정보를 출력하는 printGrades 함수를 구현하시오. 이 함수는 학생의 성적을 vector에 저장된 만큼 다음과 같이 한 줄에 한 과목씩 출력한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

5) 학생의 성적을 평균내어 평점을 구하는 getGPA 함수를 구현하시오. 이 함수는 성적 vector에 있는 A, B, C, D, F를 4, 3, 2, 1, 0으로 변환하여 모두 더해 평균을 내서 float으로 반환하는 함수이다.

* printGrades 함수 마지막 부분에 추가해서, 과목 이름과 성적 출력 후, 마지막에 평균 평점을 출력하도록 하라.

텍스트이(가) 표시된 사진

자동 생성된 설명

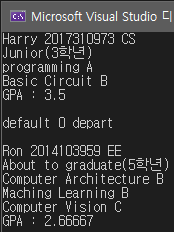
6) 학번을 보고 입력받는 해를 기준 몇 학년 인지 출력하는 getYear함수를 구현하시오. 이 함수는 year(int) 정보를 받아 Freshmen(1학년), Sophomore(2학년), Junior(3학년), Senior(4학년), About to graduate(5학년 이상) 중 하나를 출력한다.

* 휴학 등은 고려하지 않고, 입력 받은 년도와 학번의 차이로만 학년을 계산한다.

7) main 함수는 다음과 같이 작성한다.

|  |
| --- |
| int main() {  Student Harry("Harry", 2017310973, "CS");  Harry.print();  Harry.getYear(2019);  Harry.addGrade("programming", 'A');  Harry.addGrade("Basic Circuit", 'B');  Harry.printGrades();  cout << "\n\n";  Student Ron;  Ron.print();  cout << "\n";  Ron.setName("Ron");  Ron.setID(2014103959);  Ron.setDpt("EE");  Ron.print();  Ron.getYear(2019);  Ron.addGrade("Computer Architecture", 'B');  Ron.addGrade("Maching Learning", 'B');  Ron.addGrade("Computer Vision", 'C');  Ron.printGrades();  cout << "\n\n";  return 0;  } |

3-출력화면:



|  |
| --- |
| #include <string>  #include <iostream>  #include <vector>  using namespace std;  class Student {  private:  int m\_ID;  string m\_Name;  string m\_Major;  vector<string> subjects;  vector<char> grades;  public:  Student(string name, int id, string major) : m\_ID(id), m\_Name(name), m\_Major(major) {}  Student() : m\_ID(0), m\_Name("default"), m\_Major("depart") {}  void setID(int number) {  m\_ID = number;  }  void setName(string name) {  m\_Name = name;  }  void setDpt(string major) {  m\_Major = major;  }  int getNumber() {  return m\_ID;  }  string getName() {  return m\_Name;  }  string getMajor() {  return m\_Major;  }  void print() {  cout << getName() << " " << getNumber() << " " << getMajor() << endl;  }  void addGrade(string subject, char grade){  subjects.push\_back(subject);  grades.push\_back(grade);  }  void printGrades() {  for (int i=0; i < grades.size(); i++) {  cout << subjects[i] << " " << grades[i] << endl;  }  cout << "GPA : " << getGPA();  }  float getGPA() {  float sum = 0;  float GPA;  for (int i=0; i < grades.size(); i++) {  if (grades[i] == 'A') sum += 4;  if (grades[i] == 'B') sum += 3;  if (grades[i] == 'C') sum += 2;  if (grades[i] == 'D') sum += 1;  if (grades[i] == 'F') sum += 0;  }  GPA = sum / grades.size();  return GPA;  }  void getYear(int year) {  int enter = getNumber() / 1000000;  int interval = year - enter + 1;  if (interval == 1) cout << "Freshmen (1학년)" << endl;  else if (interval == 2) cout << "Sophomore (2학년)" << endl;  else if (interval == 3) cout << "Junior (3학년)" << endl;  else if (interval == 4) cout << "Senior (4학년)" << endl;  else cout << "About to grade" << '(' << interval << "학년)"<< endl;  }  };  int main() {  Student Harry("Harry", 2017310973, "CS");  Harry.print();  Harry.getYear(2019);  Harry.addGrade("programming", 'A');  Harry.addGrade("Basic Circuit", 'B');  Harry.printGrades();  cout << "\n\n";  Student Ron;  Ron.print();  cout << "\n";  Ron.setName("Ron");  Ron.setID(2014103959);  Ron.setDpt("EE");  Ron.print();  Ron.getYear(2019);  Ron.addGrade("Computer Architecture", 'B');  Ron.addGrade("Maching Learning", 'B');  Ron.addGrade("Computer Vision", 'C');  Ron.printGrades();  cout << "\n\n";  return 0;  } |
|  |