과 제 집

※ ***과제 제출 요령***

1. 과제 제출메일 : mylove**C**@gmail.com

2. 메일 제목 : 수강생 홍길동 HW1 제출

3. 제출파일명 : 해당 과제의 source파일만 제출(project 폴더를 통째로 압축하지 말 것)
   과제번호와 파일명의 번호가 일치하도록 할 것 (ex : HW1.cpp)

(HW1) 사전 프로그램 (struct 배열 이용)

다음과 같은 구조체를 선언하여 단어와 뜻을 입력 받아 5가지 메뉴(입력하기, 출력하기, 검색하기, 삭제하기, 종료)로 처리하는 프로그램을 작성하시오.
단어의 길이는 19문자 이하이며, 단어의 뜻은 79문자 이하로 입력된다고 가정한다.
사전에 저장 가능한 최대 단어의 개수는 10개이며, 단어 입력 및 삭제를 반복하더라도 10개를 초과하지 않도록 해야 한다.

(데이터 저장을 위한 구조체 사용)

```
struct Dic{
    char word[20];     // 단어를 저장하는 멤버
    char mean[80];     // 단어의 뜻을 저장하는 멤버
    int len;           // 단어의 길이를 저장하는 멤버
};
struct Dic ary[10];   <- 이곳에 단어와 뜻, 단어의 길이를 저장함
```

(입력하기 메뉴 선택 시) 단어와 뜻을 반복적으로 입력 받아 저장하다가 단어 입력란에서 "end"입력하면 입력메뉴기능을 종료하고 주 메뉴로 돌아감(입력한 단어의 개수가 10개를 초과하지 않도록 해야 함)

```
# 단어를 입력하시오 : king
# 뜻을 입력하시오 : king is king

# 단어를 입력하시오 : queen
# 뜻을 입력하시오 : queen is queen

# 단어를 입력하시오 : end      <- 종료조건 (뜻을 입력란에서 end를 입력해도 종료 함)
```

(출력하기 메뉴 선택 시) 단어기준으로 알파벳 오름차순으로 소트하여 출력함.
출력 시 단어의 뜻이 길경우 50자까지만 출력하고 뒤에 생략기호(~)를 붙여서 출력한다.
모두 출력하고 잠시 멈춘 후 아무 키나 입력하면 주 메뉴로 돌아감.

```
1.          apple( 5) : apple is apple
2.        gogumi( 6) : gogumida! gogumida!!
                :
```

(검색하기 메뉴 선택 시) 찾는 단어가 발견되면 그 단어의 뜻을 출력하고 없을 시에는 "Not found!!!" 메시지 출력하기. 반복적으로 검색하다가 검색할 단어 입력란에서 "end"입력하면 검색메뉴기능을 종료하고 주 메뉴로 돌아감

```
# 찾을 단어를 입력하시오 : queen
  단어의 뜻 : queen is queen

# 찾을 단어를 입력하시오 : prince
  해당 단어는 존재하지 않습니다.
```

찾을 단어를 입력하시오 : end <= 종료조건

(삭제하기 메뉴 선택 시) 입력된 단어를 삭제하고 없는 단어의 경우 "없는 단어입니다."라는 메세지 출력하기. 반복적으로 삭제하다가 삭제할 단어 입력란에서 "end"입력하면 삭제메뉴기능을 종료하고 주 메뉴로 돌아감.

삭제할 단어를 입력하시오 : queen
정말로 삭제하시겠습니까?(y/n) : n ← y 입력 시 삭제, n과 그 외 문자 입력 시 삭제취소
삭제가 취소되었습니다.

삭제할 단어를 입력하시오 : king
정말로 삭제하시겠습니까?(y/n) : y ← y 입력 시 삭제, n과 그 외 문자 입력 시 삭제취소
삭제되었습니다.

삭제할 단어를 입력하시오 : prince
해당 단어는 존재하지 않습니다.

삭제할 단어를 입력하시오 : end <= 종료조건

(HW2) 포인트 끝말잇기 게임 만들기

우리가 일반적으로 알고 있는 끝말잇기 게임을 약간 변형하여 재미있는 포인트 끝말잇기 게임을 만들어봅시다.

(기능 명세)

1. 점수로 계산될 5개의 포인트 단어를 hw71_pointWord.txt파일로 부터 입력 받아 오름차순으로 데이터를 소트하여 출력한다.

hw1_pointWord.txt 내용

```
tiger
hen
cow
rabbit
lion
```

2. 끝말잇기를 하는 중에 포인트 단어로 끝말을 잇게 되면 해당하는 포인트 단어가 지워지며 포인트 단어 하나가 지워질 때마다 20점의 점수를 얻게 된다.(5개의 포인트 단어를 모두 맞추면 100점을 얻게 되고 게임은 종료 됨)

3. 끝말잇기 게임은 1게임당 총 10회의 입력만 허용한다.
이때 끝말잇기 게임의 룰에 따라 이전 글자의 마지막 글자로 시작하는 단어를 적어준다.
만일 끝말잇기가 안 되는 단어를 입력했을 경우에는 "잘못 입력하셨습니다"라는 메세지를 출력한 후 재입력을 요구한다.

4. 시작단어는 "pointer"로 시작합니다.

5. 기타 자세한 실행 결과는 아래 실행 예를 참조하여서 작성한다.

(실행 예)
포인트 단어를 파일로부터 입력 받는 중입니다.... (메시지를 띄워주고 파일로부터 입력 받을 것)

* 포인트단어 : cow / hen / lion / rabbit / tiger / ← 파일로부터 입력 받은 포인트 단어를 화면에 출력
* 사용자 입력 단어 : pointer /
끝말잇기 단어 입력(1회차) : rabbit

* 포인트단어 : cow / hen / lion / tiger / ← 위에서 입력한 rabbit을 포인트단어에서 찾아 지웠으므로 출력하지 않는다.
* 사용자 입력 단어 : pointer / rabbit / ← 위에서 입력한 rabbit을 사용자 입력 단어에 추가하여 출력
끝말잇기 단어 입력(2회차) : tiger

* 포인트단어 : cow / hen / lion / ← tiger도 지워졌으므로 포인트단어란에 출력되지 않는다.
* 사용자 입력 단어 : pointer / rabbit / tiger /
끝말잇기 단어 입력(3회차) : lab ← 끝말 연결이 안되는 단어 입력 시 오류 메세지 출력
잘못 입력하셨습니다.

* 포인트단어 : cow / hen / lion /
* 사용자 입력 단어 : pointer / rabbit / tiger /
끝말잇기 단어 입력(3회차) : ribbon

* 포인트단어 : cow / hen / lion /
* 사용자 입력 단어 : pointer / rabbit / tiger / ribbon /
끝말잇기 단어 입력(4회차) : nail

* 포인트단어 : cow / hen / lion /
* 사용자 입력 단어 : pointer / rabbit / tiger / ribbon / nail /
끝말잇기 단어 입력(5회차) : lion

* 포인트단어 : cow / hen /
* 사용자 입력 단어 : pointer / rabbit / tiger / ribbon / nail / lion /
끝말잇기 단어 입력(6회차) : noel

* 포인트단어 : cow / hen /
* 사용자 입력 단어 : pointer / rabbit / tiger / ribbon / nail / lion / noel /
끝말잇기 단어 입력(7회차) : lauph

* 포인트단어 : cow / hen /

* 사용자 입력 단어 : pointer / rabbit / tiger / ribbon / nail / lion / noel / lauph /
끝말잇기 단어 입력(8회차) : hen

* 포인트단어 : cow /
* 사용자 입력 단어 : pointer / rabbit / tiger / ribbon / nail / lion / noel / lauph / hen /
끝말잇기 단어 입력(9회차) : need
:
:
* 포인트단어 : cow /
* 사용자 입력 단어 : pointer / rabbit / tiger / ribbon / nail / lion / noel / lauph / hen / need /
끝말잇기 단어 입력(10회차) : draw

** 당신의 점수는 80점 입니다

(HW3) 아파트 추첨 관리 프로그램 작성(Queue 사용)

청약자들의 데이터는 순위(1~3순위)별로 다음의 형태로 데이터 파일에 저장되어있다.

grade1.txt

일순신 ← 성명
820207 ← 생년월일
010-1234-5678 ← 연락처
:

grade2.txt

일길동
820207
010-1234-5678
:

grade3.txt

일지문덕
820207
010-1234-5678 :

추첨자 명수를 입력 받아 다음의 규칙대로 당첨자를 결정한다.

- 추첨자 명수가 10명이고 1등급 청약자가 20명이면 1등급 청약자 중 10명을 선발한다.
- 추첨자 명수가 10명이고 1등급 청약자가 7명이면 1등급 청약자 모두를 선발하고 부족한 3명은 2등급 청약자중 선발한다.

선발 시에는 난순으로 선발하며 당첨자는 win.txt파일에 저장 한 후

win.txt파일을 읽어들여 다음의 형태대로 당첨자 명단을 화면에 출력한다.

[No. 당첨자명 주민번호 연락처]

1. 이순신 820912-******* 010-9323-****
2. :

(HW4) 빙고게임 만들기

컴퓨터와 사용자의 대전용 빙고게임 구현

(세부 기능)

1. 개인연습용 빙고게임과 컴퓨터와 대전할 수 있는 빙고게임을 구현한다.
2. 빙고게임의 가로세로 크기를 입력 받아 빙고게임에 사용되는 배열은 int [N][N]; 형태의 2차원 배열을 동적메모리 할당 하여 사용하며 중복되지 않도록 1~(N*N)의 숫자로 배열을 채우고 게임을 시작한다.
   ** 이때 N값은 양수이여야 함.
3. 빙고게임에서 가로 N줄, 세로 N줄, 대각선 2줄 중에서 N개 이상의 줄이 먼저 지워지는 쪽이 승리하는 것으로 처리하되 컴퓨터와 사용자가 동시에 N개 이상 지워질 경우에는 무승부로 처리한다.
4. 게임 진행 중 이미 지워진 숫자를 입력하거나 1~(N*N)의 숫자가 아닌 값을 입력할 경우에는 재입력을 요구한다.
5. 기타 부수적인 내용은 개발자가 상황을 고려해서 작성한다.

(사용 데이터형)

- extern 변수나 배열을 사용하지 않음
- 그 외 필요한 변수 선언 : 데이터형 제한 없음

(주요 사용 기술]

- 간단한 Number select 메뉴를 이용한 User Interface 작성
- 빙고판의 내용 설정 시 난수 발생 기법 사용
- 프로그램 작성시 함수화 시킬 수 있는 부분은 최대한 함수화 시키시오.
  예) 빙고판에 숫자를 세팅하는 함수
  빙고판에 지워진 라인수 count하는 함수
  이미 지워진 난수 체크하는 함수
  메뉴 출력하고 메뉴번호 입력하는 함수 등...

(실행 예)

1. 초기메뉴 화면

```
1. 연습게임(개인 연습용)
2. 대전게임(컴퓨터와 대전용)
3. 종료

# 메뉴선택 : 1(엔터)
```

2. (1번) 연습게임 메뉴선택 시 화면 (다음은 5행 5열 빙고판을 기준으로 설명 함)

```
연습용 빙고게임을 시작합니다.

# 빙고판의 가로,세로 크기를 입력해주세요(양수값 입력) : 5(엔터)

|   9  10  20   3   1  |
|   8  21   7  15  19  |
|  22   2  13  25  24  |    <=== 빙고게임판의 내용은 1~25까지의 숫자를 난수로
|   4  11  18   5  14  |         발생시켜서 출력해준다.
|  12   6  16  23  17  |

# 지울 숫자 입력(1~25) : 1(엔터)

|   9  10  20   3   X  |
|   8  21   7  15  19  |
|  22   2  13  25  24  |  <=== 입력되어서 삭제된 숫자란의 내용은 'X'문자로 표기
|   4  11  18   5  14  |
|  12   6  16  23  17  |

# 지울 숫자 입력(1~25) : 1(엔터)
* 이미 지워진 숫자 입니다. 다시 입력하세요.
# 지울 숫자 입력(1~25) : 23(엔터)

|   9  10  20   3   X  |
|   8  21   7  15  19  |
|  22   2  13  25  24  |
|   4  11  18   5  14  |
|  12   6  16   X  17  |

(위의 작업을 5줄이 지워질 때까지 반복한 후 연습게임이 끝나고 아무 키나 치면 주메뉴로 돌아감)
```

3. (2번) 대전게임 메뉴 선택 시 화면

```
사용자:컴퓨터 대전 빙고게임을 시작합니다.

# 빙고판의 가로,세로 크기를 입력해주세요(양수값 입력) : 5(엔터)

사용자 빙고게임판 내용을 생성중입니다.
컴퓨터 빙고판 내용을 생성중 입니다.

(모든 생성이 끝나고 나면 아래와 같이 출력한다)

 [user]                          [computer]
```

```
|   9  10  20   3   1 |     |   ?   ?   ?   ?   ? |
|   8  21   7  15  19 |     |   ?   ?   ?   ?   ? |  <<=== 사용자의 숫자는
|  22   2  13  25  24 |     |   ?   ?   ?   ?   ? |     값으로 출력하고 컴퓨터
|   4  11  18   5  14 |     |   ?   ?   ?   ?   ? |     의 숫자는 ?로 출력
|  12   6  16  23  17 |     |   ?   ?   ?   ?   ? |

# 지울 숫자 입력(1~25) : 23(엔터)  <<===  이미 지운 숫자 입력시 재입력 요구

[user]                          [computer]
|   9  10  20   3   1 |     |   ?   ?   ?   ?   ? |
|   8  21   7  15  19 |     |   ?   ?   ?   ?   ? |
|  22   2  13  25  24 |     |   ?   ?   X   ?   ? |  <<=== 사용자가 입력한
|   4  11  18   5  14 |     |   ?   ?   ?   ?   ? |        숫자를 지움
|  12   6  16   X  17 |     |   ?   ?   ?   ?   ? |

# 컴퓨터가 선택한 숫자는 (23)입니다.    <<=== 컴퓨터의 숫자는 난수를 발생시켜서 처리함.
이때 이미 지운 숫자가 난수로 발생하면 다시 난수를 발생시켜야 함

# 컴퓨터가 선택한 숫자는 (9)입니다.

[user]                          [computer]
|   X  10  20   3   1 |     |   ?   ?   X   ?   ? |
|   8  21   7  15  19 |     |   ?   ?   ?   ?   ? |
|  22   2  13  25  24 |     |   ?   ?   X   ?   ? |  <<=== 컴퓨터가 발생시킨
|   4  11  18   5  14 |     |   ?   ?   ?   ?   ? |        숫자를 지움
|  12   6  16   X  17 |     |   ?   ?   ?   ?   ? |

(위의 작업을 반복적으로 수행하며 사용자나 컴퓨터 둘 중 어느 한쪽이라도 먼저 N줄 이상이
지워지면 게임종료)
 :
 :
# 사용자 승!  (또는 컴퓨터 승!  또는 사용자, 컴퓨터 무승부! 의 형태로 게임 최종결과
출력하기)

# 아무 키나 치면 주 메뉴로 돌아갑니다.
```

[C++ 과제]

(HW5)

이름과 세 과목의 점수를 키보드로부터 입력 받아 총점과 평균을 구하여 출력한다. 평균은 소수점 이하 둘째 자리까지 출력한다.

실행결과

```
이름 : 홍 길동(엔터)
세 과목의 점수 : 70 90 88(엔터)

                    =====< 성적표 >=====
============================================================
      이름       국어       영어       수학       총점       평균
============================================================
    홍 길동         70         90         88        248      82.67
```

(HW6)

키보드로부터 문자열을 입력 받아 문자열의 길이를 출력한다. 이어서 하나의 문자를 입력 받고 먼저 입력 받은 문자열에서 새로 입력 받은 문자가 나오기 전까지의 길이를 구해서 출력한다. 문자열의 길이를 구하는 함수를 하나만 작성하되 문자를 처리하는 매개변수는 기본 전달인자를 사용하며, 함수명은 my_strlen으로 하고 길이를 구하여 return 하는 함수로 작성한다.
단, 없은 문자 입력 시 전체 문자열 출력

실행결과

```
문자열 입력 : strawberry(엔터)
문자열의 길이 : 10
문자 입력 : b(엔터)
b 이전 까지의 길이 : 5
```

(HW7)

저금통에 잔돈을 저금하는 프로그램을 작성한다. 키보드로부터 동전의 금액과 개수를 반복적으로 입력하다가 입력이 끝나면 총 저축액을 출력한다. 동전의 단위는 500원, 100원, 50원, 10원으로 하고 동전의 금액으로 문자가 입력되면 입력을 종료한다. 각 동전의 개수를 저장하기 위해 Savings구조체 변수를 사용하며, 제시한 네 개의 함수를 작성하여 활용한다.

```
struct Savings{
    int w500;   // 500원짜리 동전의 개수 저장
    int w100;   // 100원짜리 동전의 개수 저장
    int w50;   // 50원짜리 동전의 개수 저장
    int w10;   // 10원짜리 동전의 개수 저장
};

void init(Savings &s);   // 저금통의 초기화
  : Savings 구조체내의 4개의 멤버를 모두 0으로 초기화 함

void input(int &unit, int &cnt);   // 동전금액, 동전 개수를 저장할 변수
  : 키보드로부터 동전금액과 동전 개수를 입력 받아  unit과 cnt에 저장함
    이 때 잘못된 동전금액이나 문자가 입력되면 재 입력 할 것 (입력 예외처리)

void save(Savings &s, int unit, int cnt);   // 저금통, 단위, 개수를 전달
  : 저금통 구조체 변수의 멤버를 전달 받은 동전의 개수 만큼 누적함

int total(Savings &s);
  : 저금통의 총 저축액을 계산하여 리턴하는 함수
```

실행결과

```
동전의 금액 : 500(엔터)
동전의 개수 : 7(엔터)
동전의 금액 : 50(엔터)
동전의 개수 : 3(엔터)
동전의 금액 : 500(엔터)
동전의 개수 : 2(엔터)
동전의 금액 : 100(엔터)
동전의 개수 : 15(엔터)
동전의 금액 : 10(엔터)
동전의 개수 : 4(엔터)
동전의 금액 : #(엔터)
총 저금액 : 6190
```

(HW8)

포인터변수가 연결하고 있는 두 개의 문자열 출력하고 포인터변수의 값을 서로 바꾼 후에 다시 한번 두 문자열을 출력한다. 포인터변수의 값을 바꾸는 함수는 참조매개변수를 사용하며 이름은 swap_ptr 으로 작성한다. (swap_ptr()함수의 parameter로 포인터 변수를 사용하지 말 것)

```
int main()
{
    char *ap="apple";
    char *bp="banana";

    cout << "바꾸기 전의 문자열 : " << ap << "  " << bp << endl;
    swap_ptr(ap, bp);
    cout << "바꾼 후의 문자열 : "<< ap << "  " << bp << endl;

    return 0;
}
```

실행결과

```
바꾸기 전의 문자열 : apple banana
바꾼 후의 문자열 : banana apple
```

(HW9)

유통기한에 따라 제품의 가격을 할인 판매하는 프로그램을 작성한다. 키보드로부터 품명, 정가, 유통기한을 입력하면 현재 날짜로부터 유통기한까지 남은 일수를 계산하고, 그에 따라 할인율을 차등 적용하여 정가를 수정한다.

할인율은 다음과 같이 적용하고 제시한 구조체와 함수들을 활용하여 작성한다.

- 유통기한 0~3일 전까지 50%, 4~10일 전까지 20% 할인판매하고, 그 외는 정가판매
- 유통기한이 지난 경우는 "유통기한이 지났습니다!" 메시지 출력

```
struct Goods
{
    char item[50];   // 품목
    int price;   // 정가
    int year;   // 유통기한(년도)
    int mon;   // 유통기한(월)
    int day;   // 유통기한(일)
    int discount;   // 할인율
};
void input(Goods &s);            // 품목, 정가, 유통기한 입력 함수
void selling_price(Goods &s);    // 유통기한에 따라 할인율 결정
void prn(const Goods &s);        // 품목, 판매가, 유통기한, 할인율 출력 함수
int tot_days(int y, int m, int d);   // 년,월,일로 총일수 계산
int leap_check(int year);            // 해당 년도가 윤년인지 검사하는 함수
```

실행결과

```
품목 입력 : 철이네 생생두부(엔터)
정가 입력 : 2300(엔터)
유통기한 입력 : 2018 1 30(엔터)
품명 : 철이네 생생두부
판매가 : 1150
유통기한 : 2018년 1월 30일
할인율 : 50%
```

** 프로그램을 실행시킨 날짜 : 2018년 1월 28일

** system으로부터 년, 월, 일 입력하는 함수는 자료게시판 참고

(HW 10) 시간을 저장하고 출력하는 프로그램을 작성한다.

Time객체는 시간과 분을 따로 저장하며 분은 항상 0~59 사이의 값을 갖는다.
객체를 선언할 때 초기화를 하지 않으면 0시간 0분으로 객체를 생성하며 초기화를 할 때는 시간과 분으로 초기화를 한다.
또한 실수값으로도 초기화를 할 수 있도록 한다. 예를 들어, 4.5로 초기화하면 4시간 30분이 된다.
다음의 클래스와 출력결과를 참고하여 멤버함수의 정의를 완성한다.

```
class Time{
private:
    int hour;
    int min;
public:
    Time();                 // 디폴트 생성자 – 0시 0분으로 초기화
    Time(int, int);         // 시간, 분으로 초기화하는 생성자
    Time(double);           // 시간을 실수값으로 초기화하는 생성자
    Time(const Time &tr);   // 복사 생성자
    Time plus(const Time &tr);  // 두 객체의 시간을 더해서 리턴한다.
    void setHour(int);      // hour멤버의 값을 전달인자로 수정하는 함수
    int getHour();          // hour멤버의 값을 리턴하는 함수
    void setMin(int);       // min멤버의 값을 전달인자로 수정하는 함수
    int getMin();           // min멤버의 값을 리턴하는 함수
};
// 여기에 각 멤버함수의 정의를 작성한다.

int main()
{
    Time a(3, 20), b;
    cout << a.getHour() << "시간" << a.getMin() << "분"<< endl;
```

```
    b.setHour(4);
    b.setMin(42);
    cout << b.getHour() << "시간" << b.getMin() <<"분"<< endl;
    Time res=a.plus(b);
    cout << "두 시간을 더하면 : " << res.getHour() << "시간" << res.getMin() << "분" << endl;
    return 0;
}
```

실행결과

```
3시간 20분
4시간 42분
두 시간을 더하면 : 8시간 2분
```

(HW 11) 로봇 일시키기 프로그램

```
class Robot
{
private:
        char *name;
        int energy;
        void errPrn(); // 에너지 부족- 에러메시지 출력
public:
        Robot();   // name은 Null string으로, energy는 0으로 초기화
        ~Robot();
        Robot(char *name, int energy=0);
        Robot(Robot& r);
        void go();                  // 전진 메시지 출력 후 에너지 10 감소
        void back();                // 후진 메시지 출력 후 에너지 20 감소
        void turn();                // 턴 메시지 출력 후 에너지 30 감소
        void jump();                // 점프 메시지 출력 후 에너지 40 감소
        void charge(int e);         // e값 만큼 충전
        char* getName();            // name멤버의 값 리턴
        void setName(char *);       // name멤버의 값을 전달된 문자열로 재 초기화
        int getEnergy();            // energy멤버의 값 리턴
        void setEnergy(int);        // energy멤버의 값을 전달된 숫자로 재 초기화
};
```

위의 Robot 클래스를 이용하여 다음의 프로그램을 작성하시오.

(기능)

1. 구입할 로봇의 대수를 입력 받아 로봇대수만큼 Robot객체를 생성하여 사용하자.
   구입할 로봇 대수를 입력하시오 : 3(enter)

2. 구입한 로봇 대수만큼 순차적으로 로봇의 이름과 초기 에너지량을 입력 받아 초기화 한 후 일을 시키자.
   1번 로봇명을 입력하시오 : 짱가(enter)
   짱가의 에너지 양을 입력하시오 : 100(enter)

   2번 로봇명을 입력하시오 : 태권브이(enter)
   태권브이의 에너지 양을 입력하시오 : 200(enter)

   3번 로봇명을 입력하시오 : 마징가(enter)

마징가의 에너지 양을 입력하시오 : 150(enter)

로봇명 선택(짱가, 태권브이, 마징가) : 태권브이(enter)
할일 선택(1.전진/2.후진/3.회전/4.점프/5.충전) : 2(enter)
태권브이 후진...

로봇명 선택(짱가, 태권브이, 마징가) : 마징가(enter)
할일 선택(1.전진/2.후진/3.회전/4.점프/5.충전) : 4(enter)
마징가 점프...

로봇명 선택(짱가, 태권브이, 마징가) : 짱가(enter)
할일 선택(1.전진/2.후진/3.회전/4.점프/5.충전) : 5(enter)
충전할 에너지양 입력 : 30(enter)
짱가 에너지 30 충전...

: (기타등등 일을 시킴)

로봇명 선택(짱가, 태권브이, 마징가) : (enter) <--- 종료조건 : 엔터키 입력 시 종료

3. 로봇 일 시키기가 끝나면 로봇들의 남은 에너지량을 화면에 출력하고 종료한다.
1. 짱가 : 50
2. 태권브이 : 70
3. 마징가 : 20

(HW 12)

S은행에 적금을 관리하기 위해 만든 Save 클래스를 이용해서 세율에 따른 이자소득과 총 수령액을 계산하여 출력하는 프로그램을 작성하시오.

다음의 클래스와 실행결과를 참고하여 나머지 부분을 완성한다.

```
class Save
{
private:
        char name[20];   // 이름
        int capital;   // 원금
        int profit;   // 이자소득
        int tax;   // 세금
        int tot;   // 총금액
        static double ratio;   // 이율 – 기본 이율은 원금의 20%로 설정하시오
public:
        static double tax_ratio;   // 세율 – 기본 세율은 10%로 설정하시오
        Save(const char *np="아무개", int n=0);   // 오버로디드 생성자
        void calculate();   // 총금액
        static void set_ratio(double r); // 정적 멤버함수 :  ratio를 수정하기 위한 멤버함수
        //getter, setter함수 선언
};
int main()
{
        Save   a("철이", 1000000), b("메텔", 2000000);
        a.calculate();
        cout << a << endl;
        Save::tax_ratio=0.15;   // 세율 변경
        Save::set_ratio(0.25);   // 이율 변경
        b.calculate();
        cout << b << endl;
        return 0;
}
```

실행결과

```
이름 : 철이
원금 : 1000000
이자소득 : 200000
세금 : 20000
총금액 : 1180000
이율 : 0.2
세율 : 0.1

이름 : 메텔
원금 : 2000000
이자소득 : 500000
세금 : 75000
총금액 : 2425000
이율 : 0.25
세율 : 0.15
```

(HW 13)

시간을 저장하는 Time클래스에 정적 데이터멤버와 정적 멤버함수를 추가하여 시간이 출력되는 방식을 바꿀 수 있도록 프로그램을 작성한다. 정적 데이터멤버 mode는 시간이 출력되는 방식을 결정한다. 즉, mode가 열거상수 integer이면 출력 멤버함수는 시간과 분의 형태로 출력하고 real이면 실수값 형태로 출력한다. 초기값은 integer로 설정한다. 정적 멤버함수 mode_change는 mode의 값을 전환하는 기능을 수행한다. 즉, mode가 integer일 때 호출되면 real로 바꾸고 real이었다면 integer로 바꾼다. 다음의 클래스와 실행결과를 참고하여 나머지 부분을 완성한다.

```
class Time{
private:
    int hour;
    int min;
    double time;  // 시간을 실수값 형태로 저장
    static int mode;  // 출력 모드를 저장하는 정적 데이터멤버
    enum {integer, real};  // 열거상수 정의
public:
    Time();  // 디폴트 생성자
    Time(int, int);  // 시간, 분으로 초기화하는 생성자
    Time(double);  // 시간을 실수값으로 초기화하는 생성자
    Time(const Time &tr);  // 복사 생성자
    Time plus(const Time &tr);  // 두 객체의 시간을 더해서 리턴한다.
    void setHour(int);  // hour멤버의 값을 전달인자로 수정하는 함수
    int getHour();  // hour멤버의 값을 리턴하는 함수
    void setMin(int);  // min멤버의 값을 전달인자로 수정하는 함수
    int getMin();  // min멤버의 값을 리턴하는 함수
    friend ostream &operator<<(ostream &os, const Time &br); // 출력 함수
    static void mode_change();  // mode를 바꾸는 정적 멤버함수
};
int main()
{
    Time a(3, 21);  // 3시간 21분으로 객체생성
    Time b(2.7);  // 2.7시간으로 객체생성
    cout << "기본 출력 형태(시간, 분)..." << endl;
    cout << a << endl;
    cout << b << endl;
    Time::mode_change();  // 시간을 출력하는 모드를 바꾼다.
```

```
    cout << "출력모드를 바꾼 후..." << endl;
    cout << a << endl;
    cout << b << endl;
    return 0;
}
```

실행결과

```
기본 출력 형태(시간, 분)...
3시간 21분
2시간 42분
출력모드를 바꾼 후...
3.35시간
2.7시간
```

(HW 14)

5개의 과일이름을 키보드로부터 입력 받아서 문자열을 저장하는 MyString클래스의 객체배열에 차례로 저장한다. 그리고 첫 번째 문자열과 두 번째 문자열을 관계연산자로 비교하여 길이가 긴 문자열을 출력한다. 문자열을 출력할 때는 문자열의 길이를 괄호와 함께 출력한다. 그 후에 첫 번째 문자열 뒤에 나머지 모든 문자열을 이어서 저장하고 모든 문자열을 출력한다. 다음의 클래스 선언과 메인함수, 출력결과 등을 참고하여 프로그램을 완성한다.

```
class MyString{
private:
    char *str;  // 문자열을 연결할 포인터멤버
    int len;  // 문자열의 길이를 저장
public:
    MyString();  // 디폴트 생성자
    MyString(const char *cp);  // 오버로디드 생성자
    MyString(const MyString &br);  // 복사 생성자
    ~MyString();  // 소멸자
    MyString &operator=(const MyString &br);  // 대입연산자 멤버함수
    MyString operator+(const MyString &br);  // 덧셈 연산자
    bool operator>(const MyString &br);  // 관계연산자 멤버함수(길이 비교)
    void operator<<(ostream &os);  // 출력연산자 멤버함수
    void operator>>(istream &is);  // 입력연산자 멤버함수
};
ostream &operator<<(ostream &os, MyString &br);  // 출력 일반함수
istream &operator>>(istream &is, MyString &br);  // 입력 일반함수

int main()
{
    MyString ary[5];  // 객체 배열
    MyString res;  // 문자열을 모두 붙여서 저장할 객체
    int i;
    cout << "5개의 과일이름 입력 : ";
    for(i=0; i<5; i++){
        cin >> ary[i];  // 입력연산자 사용
    }
    cout << "첫번째와 두번째 중 긴 과일 이름 : ";
    if(ary[0]>ary[1]){  // 관계연산자 사용
        cout << ary[0] << endl;  // 출력연산자 사용
    }
    else{
        cout << ary[1] << endl;
    }
    res=ary[0]+ary[1]+ary[2]+ary[3]+ary[4]; // 덧셈연산자 사용
```

```
    cout << "모든 문자열 출력:" << res << endl;
    cout << "배열내의 문자열 출력..." << endl;
    for(i=0; i<5; i++){
        cout << ary[i] << endl;
    }
    return 0;
}
```

실행결과

```
5개의 과일이름 입력 : apple banana kiwi melon orange( 엔터)
첫번째와 두번째 중 길 과일 이름 : banana(6)
모든 문자열 출력 : applebananakiwimelonorange(26)
배열내의 문자열 출력...
apple(5)
banana(6)
kiwi(4)
melon(5)
orange(6)
```

(HW 15)

피트니스 클럽의 회원관리 프로그램을 다음의 조건에 맞게 기능을 작성하시오.

- 클럽회원의 등록 가능한 회원수는 최대 100명으로 제한한다.
  Fitness * fary[100]; 의 형태로 Fitness * 변수 100개로 이루어진 배열을 선언한 후
  한 명의 회원이 가입할 때 마다 Fitness 객체를 동적할당하여 fary배열의 각 방에 순서대로 붙여넣는다. 이때 전체회원이 몇 명인지를 관리하기 위해서는 memberCnt변수를 사용하시오.

- 사용메뉴는 다음과 같다.
  1. 회원등록 // 회원 등록 기능
  2. 회원전체보기 // 한 줄에 한 명씩 전체 회원정보 출력하기
  3. 회원정보검색 // 회원명에 대한 개인정보를 출력한다.
  4. 회원탈퇴 // 회원번호에 해당하는 회원 객체 삭제
  5. 특별관리 회원 // bmi를 검사하여 고도비만, 비만, 과체중에 해당하는 회원 정보 출력
  6. 종료 // 종료할 때 할당했던 모든 객체 free 시킴

  - BMI 계산법

    BMI = 몸무게(kg) / (키(m)의 제곱) // 단, 키는 단위가 미터이다.
    * BMI 수치에 따른 소견표
    저체중 : 18.5 미만 정상체중 : 18.5~25.0 미만
    과체중 : 25.0~30.0 미만 비만 : 30.0~40.0 미만
    고도비만 : 40.0 이상

다음의 Fitness 클래스를 이용하여 프로그램을 작성한다.

```
class Fitness
{
  private :
     int num;            // 회원번호
     char *name;         // 성명
     double weight;      // 몸무게(kg단위)
     double height;      // 신장(키-cm단위)

  public :
     // 생성자 함수, 소멸자 함수
```

```
    Fitness();//회원번호 0, 성명 null string, 몸무게 0kg, 신장 0cm로 초기화
    Fitness(int num, char *name, double weight, double height);
    Fitness(Fitness &r);
    ~Fitness();       // 동적 할당된 메모리 해제
  // 연산자 함수
    Fitness & operator=(Fitness &r);
    bool operator==(Fitness &r);
  // setter, getter 함수
    void setNum(int num);              // num 멤버 초기화 함수
    int getNum();                      // num 멤버값 리턴 하는 함수
    void setName(char *name);          // name 멤버 초기화 함수
    char * getName();                  // name 멤버값 리턴 하는 함수
    void setWeight(double weight);     // weight 멤버 초기화 함수
    double getWeight();                // weight 멤버값 리턴 하는 함수
    void setHeight(double height);     // height 멤버 초기화 함수
    double getHeight();                // height 멤버값 리턴 하는 함수
  // 기타멤버함수
    double bmi();              // 해당 회원의 bmi를 계산하여 리턴 하는 함수
};
```

(HW 16)

철이와 메텔이 만원의 행복을 촬영한다. 각각 10000원의 잔액을 가지고 시작하며,
사용내역과 금액을 입력한 후에 잔액이 많은 사람의 데이터를 출력한다.
사용내역은 100개 이내로 하며 다음의 파일과 실행결과를 참고하여 작성한다.

(Happy.h)

```
class Happy
{
private:
    string name;      // 이름
    int money;        // 잔액
    char *list[100];  // 사용 내역을 저장하는 char * 배열
    int index;        // 사용 내역 개수
public:
    Happy(string np="", int money=10000);  // 디폴트 생성자 함수
    Happy(Happy &r);       // 복사 생성자 함수
    ~Happy();              // 소멸자 함수
    Happy & operator=(Happy &r);    // 대입연산자 함수
    void use(char *lp, int n);  // 사용 함수(사용내역과 금액을 받는다)
    Happy &winner(Happy &r); // 잔액 비교하여 많은 금액이 남은 사람의 객체리턴
    void setName(string &name);
    string & getName();
    void setMoney(int money);
    int getMoney();
    char ** getList();
    int getIndex();
};
```

(main.cpp)

```
int main()
{
	Happy a("철이"), b("메텔"), w;
	char item[100];
	int price;

	cout << "철이가 돈을 씁니다..." << endl;
	while(1){
		cout << "구입 내역 : ";
		cin >> item;
		if(strcmp(item, "끝")==0) break;
		cout << "금액 입력 : ";
		cin >> price;
		a.use(item, price);
	}

	cout << "메텔이 돈을 씁니다..." << endl;
	while(1){
		cout << "구입 내역 : ";
		cin >> item;
		if(strcmp(item, "끝")==0) break;
		cout << "금액 입력 : ";
		cin >> price;
		b.use(item, price);
	}

	cout << "최종 승자는?" << endl;
	w=a.winner(b);
	cout << w.getName().getStr() << "이고 총 " << w.getIndex()
		<< "건을 사용하고 남은 금액은 " << w.getMoney() << "원 입니다.";

	// 승자의 사용내역을 출력하는 코드 넣어서 완성하시오.

	return 0;
}
```

(실행결과)

```
철이가 돈을 씁니다...
구입 내역 : 만두(엔터)
금액 입력 : 1200(엔터)
구입 내역 : 델리명가밥(엔터)
금액 입력 : 1700(엔터)
구입 내역 : 불우이웃돕기(엔터)
```

```
금액 입력 : 3000(엔터)
구입 내역 : 끝(엔터)

메텔이 돈을 씁니다...
구입 내역 : 털모자(엔터)
금액 입력 : 700(엔터)
구입 내역 : 자외선차단제(엔터)
금액 입력 : 4800(엔터)
구입 내역 : 머리끈(엔터)
금액 입력 : 200(엔터)
구입 내역 : 끝(엔터)

최종 승자는?
이름 : 메텔이고 총 3건을 사용하고 남은 근액은 4300원 입니다.
사용 내역 : 털모자 자외선차단제 머리끈
```

<<<=== 이 부분도 출력되게 main()함수 끝 부분에 코드를 추가하시오.

(HW 17) 배열 기본 클래스 작성

배열은 기본적으로 포인터의 계산에 의해 기억공간을 참조한다. 따라서 유효하지 않은 첨자(index)를 사용하더라도 컴파일 단계에서 오류를 잡아내지 못하고 실행할 때 문제를 일으킨다.
또한 대입연산과 같은 기본적인 연산 자체가 불가능하다. 이제 내장 배열보다 융통성 있고 활용도가 높은 배열클래스를 만들어보자. 배열클래스는 int형 배열을 동적으로 구현하자.
그리고 세 과목의 점수를 저장할 객체를 생성하여 점수를 입력 받고 총점과 평균을 구해 출력한다. 평균은 소수점 이하 둘째 자리까지 출력한다. 다음의 클래스 선언과 출력결과를 참고한다. 간단한 멤버함수는 클래스 안에 정의하였다.

```
class MyArray
{
private:
    int *ary;
    int size;
public:
    MyArray() { ary=0; size=0; }                    // 디폴트 생성자
    MyArray(int s);                                 // 오버로디드 생성자
    MyArray(const MyArray &br);                     // 복사 생성자
    ~MyArray() { delete [] ary; }                   // 소멸자
    MyArray &operator=(const MyArray &br);   // 대입연산자
    bool put(int index, int  val);    // val의 값을 index방에 저장. 성공하면 true, 실패하면 false 리턴
    bool get(int index, int &getdata);  // 배열요소 중 index방의 값을 getdata에 저장
                                        // 성공하면 true, 실패하면 false 리턴
    int getSize();                      // 배열요소의 개수 확인
    int *getAry(void);                  // 배열요소의 내용 확인
};

    #include <iostream>
    using namespace std;
    #include "MyAry.h"
    int main()
    {
        cout.setf(ios::fixed, ios::floatfield);
        cout.precision(2);
```

```
        MyArray score(3);
        int i;
        bool res;
        int num;
        int tot=0;
        int temp;

        for(i=0; i<4; i++){
                cout << i+1 << "번 학생의 성적 입력 : ";
                cin >> temp;
                if(score.put(i, temp)==false)
                {
                        cout << "배열 쓰기 실패" <<endl;
                }
        }
        for(i=0; i<4; i++){
                res = score.get(i, num);
                if(res)
                {
                        tot+=num;
                }
                else
                {
                        cout << "배열 읽기 실패" << endl;
                }
        }
        cout << "평균   : " << tot/3.0 << endl;

        MyArray tempArray(score);
        MyArray tmp;
        tmp=score;

        return 0;
    }
```

(HW 18) 배열 파생 클래스 작성

HW17에서 작성한 MyArray 클래스를 상속받아 좀더 기능이 추가된 클래스를 설계해보자. 파생클래스는 배열에 저장한 값 중에서 최고값을 저장하는 멤버를 추가하고 새로운 데이터가 입력될 때마다 항상 최고값이 유지되도록 put함수를 재정의한다. 그리고 두 객체의 최고값을 비교하기 위해 비교연산자 멤버함수를 추가한다. 다음의 클래스와 실행결과를 참고하여 작성한다.

```
class SmartArray : public MyArray
{
private:
    int max;   // 최고값 저장
public:
    SmartArray() {}   // 디폴트 생성자
    SmartArray(int s) : MyArray(s), max(0) // 오버로디드 생성자
    {}
    bool put(int index, int val);   // put함수의 재정의
    int getMax()                    // max값 리턴
```

```
    bool operator>(const SmartArray &br);   // 비교연산자
};
```

실행결과

```
철이의 세 과목의 점수 입력 : 70 80 95
철이의 총점 : 245
철이의 평균 : 81.67
메텔의 다섯 과목의 점수 입력 : 20 32 15 45 99
메텔의 총점 : 211
메텔의 평균 : 42.20
철이는 최고점은 메텔보다 크지 않다...
```

(연습문제 19) 배열 클래스를 템플릿 클래스로 작성하시오

HW17에서 작성한 MyArray를 템플릿 클래스로 작성하여 다음의 main()함수로 테스트를 하여보자

(testMain.cpp)

```
#include <iostream>
using namespace std;
#include "MyArray.h"
int main()
{
        cout.setf(ios::fixed, ios::floatfield);
        cout.precision(2);
        MyArray<double> height(3);

        int i;
        bool res;
        double dnum;
        double tot=0;
        double temp;
        cout << "세 학생의 키 입력 : ";
        for(i=0; i<3; i++){
                cin >> temp;
                height.put(i, temp);
        }
        for(i=0; i<3; i++){
                res = height.get(i, dnum);
                if(res)
                {
                        tot+=dnum;
                }
                else
```

```
                {
                        cout << “배열 읽기 실패” << endl;
                }
        }
        cout << "평균키 : " << tot/3.0 << endl;
        return 0;
}
```

(연습문제 20) MyArray 템플릿 클래스로부터 상속받은 SmartArray 구현하기

```
#include "MyArray.h"
#include <iostream>
using namespace std;

template <typename T>
class SmartArray : public MyArray<T>
{
private:
        T max;
public:
        SmartArray() {}
        SmartArray(int s) : MyArray<T>(s), max(0) {}
        bool put(int index, T val);
        T getMax();
        bool operator>(const SmartArray<T> &br);
};

#include <iostream>
using namespace std;
#include "SmartArray.h"
int main()
{
        cout.setf(ios::fixed, ios::floatfield);
        cout.precision(2);
        SmartArray<double> height(3);

        int i;
        bool res;
        double dnum;
        double tot=0;
        double temp;
        cout << "세 학생의 키 입력 : ";
        for(i=0; i<3; i++){
                cin >> temp;
```

```
                height.put(i, temp);
        }
        for(i=0; i<3; i++){
                res = height.get(i, dnum);
                if(res)
                {
                        tot+=dnum;
                }
                else
                {
                        cout << “배열 읽기 실패” << endl;
                }
        }
        cout << "평균키 : " << tot/3.0 << endl;
        return 0;
}
```

(HW21) 문자열내의 2진 비트열 패턴 검사

1. 문자열로 저장되어 제공되는 2진 비트열의 패턴의 종류와 각 패턴 별 발생 빈도수를 조사하는 프로그램 작성
2. 패턴의 길이는 사용자에게 입력 받아 검사한다.(패턴의 길이는 1~7 까지로 제한 됨)
3. 패턴이 몇 종류일지 예측할 수 없으므로 double linkedlist를 이용하여 검사한 패턴을 저장한다.
4. 패턴검사 후 출력 시 패턴문자열이 오름차순이 되도록 sort하여 발생 빈도수와 함께 출력한다.

(patternData.txt 파일 내용)
2진 비트열 3줄이 저장 되어 있음 (비트열의 길이는 최대 100바이트를 초과하지 않음)

```
11101100011101101010111101
000100101000011111010101111111101101
10100111101011011110000101011101101010100000111
```

(실행 결과 화면)

*** 검사할 패턴의 길이를 입력하시오 : 4(엔터)**

```
** 비트열 : [ 11101100011101101010111101 ]의 검사 결과 **
[ 0001 ] : 1개
[ 0011 ] : 1개
[ 0101 ] : 2개
[ 0110 ] : 2개
[ 0111 ] : 2개
[ 1000 ] : 1개
[ 1010 ] : 2개
[ 1011 ] : 3개
[ 1100 ] : 1개
[ 1101 ] : 4개
[ 1110 ] : 3개
[ 1111 ] : 1개
---------------------------------------------------------------------
** 비트열 : [ 000100101000011111010101111111101101 ]의 검사 결과 **
```

```
[ 0000 ] : 1개
[ 0001 ] : 2개
[ 0010 ] : 2개
[ 0011 ] : 1개
[ 0100 ] : 2개
[ 0101 ] : 3개
[ 0110 ] : 1개
[ 0111 ] : 2개
[ 1000 ] : 1개
[ 1001 ] : 1개
[ 1010 ] : 3개
[ 1011 ] : 2개
[ 1101 ] : 3개
[ 1110 ] : 2개
[ 1111 ] : 7개
------------------------------------------------------------------------
** 비트열 : [ 10100111101011011110000101011101101010100000111 ]의 검사 결과 **
[ 0000 ] : 2개
[ 0001 ] : 2개
[ 0010 ] : 1개
[ 0011 ] : 2개
[ 0100 ] : 2개
[ 0101 ] : 5개
[ 0110 ] : 2개
[ 0111 ] : 4개
[ 1000 ] : 2개
[ 1001 ] : 1개
[ 1010 ] : 6개
[ 1011 ] : 4개
[ 1100 ] : 1개
[ 1101 ] : 4개
[ 1110 ] : 3개
[ 1111 ] : 2개
------------------------------------------------------------------------
```

*** 검사할 패턴의 길이를 입력하시오 : 2**

```
비트열 :[11101100011101101010111101]의 검사결과
[00] :  2개
[01] :  7개
[10] :  7개
[11] :  9개
------------------------------------------------------------------------
비트열 :[000100101000011111101010111111101101]의 검사결과
[00] :  6개
[01] :  9개
[10] :  8개
[11] : 12개
---------------------------------------- ------------------------------
비트열 :[10100111101011011110000101011101101010100000111]의 검사결과
[00] :  7개
[01] : 13개
[10] : 13개
[11] : 12개
------------------------------------------------------------------------
```

*** 검사할 패턴의 길이를 입력하시오 : 3**

```
비트열 :[11101100011101101010111101]의 검사결과
[000] :  1개
[001] :  1개
[010] :  2개
[011] :  4개
[100] :  1개
[101] :  6개
[110] :  5개
[111] :  4개
--------------------------------------------------------------------------
비트열 :[00010010100001111101010111111101101]의 검사결과
[000] :  3개
[001] :  3개
[010] :  5개
[011] :  3개
[100] :  2개
[101] :  6개
[110] :  3개
[111] :  9개
--------------------------------------------------------------------------
비트열 :[10100111101011011110000101011101101010100000111]의 검사결과
[000] :  4개
[001] :  3개
[010] :  7개
[011] :  6개
[100] :  3개
[101] : 10개
[110] :  5개
[111] :  6개
--------------------------------------------------------------------------
```