Exam #2

경희대학교
KYUNG HEE UNIVERSITY

시험 안내

경희대학교 KYUNG HEE UNIVERSITY

- **주의 사항**
  - 참고할 자료들을 미리 다운로드 받은 후, 시험 시작 5분 전 네트워크 차단
  - 휴대폰 등의 전자기기를 꺼낼 시 부정행위로 간주
- **업 로드 방법**
  - 실습/과제와 동일한 방법으로 솔루션 구성
    - 전체 시험을 솔루션 하나로 구성. 솔루션 이름은 exam02
    - 각 문제를 하나의 프로젝트로 구성. 프로젝트 이름은 QuestionX (X는 문제 번호. 예: Question2)
  - http://dke.khu.ac.kr/aoop 의 [Assignments Submission] 게시판에 업 로드
  - 파일명과 게시물 제목을 아래와 같이 통일
    - 포맷: [분반]_ex02_학번
    - 예: [A]_ex02_2012345678

정수를 배열처럼 저장할 수 있는 정수 배열 클래스 IntArray를 작성하시오.

1. private 멤버 변수

int *m_pArr : 정수의 배열을 동적으로 할당하기 위한 포인터
int m_nSize; 배열의 크기를 나타냄.

2. 생성자/소멸자

IntArray(void) : 크기가 0인 빈 배열을 생성
Initialization List를 이용하여 m_pArr = NULL, m_nSize = 0 초기화
IntArray(int nSize) : 배열 크기가 nSize인 배열을 생성
Initialization List를 이용하여 m_nSize = nSize 초기화
m_pArr에 nSize 크기의 공간을 할당하고 각 값을 0으로 초기화
IntArray(const IntArray& arr) : 전달받은 객체를 이용하여 객체 복사 생성
~IntArray(void) : m_pArr에 메모리가 할당된 경우 메모리를 해제

3. Friend로 구현해야 할 연산자

+ 연산자 : 앞 배열에 뒤 배열을 연결한 새로운 배열을 생성하는 이항 연산자

```
friend IntArray operator+(const IntArray& lhs, const IntArray& rhs);
```

앞/뒤 배열의 내용이 변경되면 안됨.

<< 연산자 : cout으로 배열을 출력하기 위한 이항 연산자

```
friend std::ostream& operator<<(std::ostream& out, const IntArray& rhs);
```

4. 클래스 멤버로 오버로딩해야 할 연산자

= 연산자 : 피연산자로 입력된 배열과 동일한 구성을 갖도록 대입하는 연산자

IntArray& operator=(const IntArray& rhs);

+= 연산자 : 각 자리의 원소값과 피연산자로 입력된 배열의 각 자리의 원소값을 더하여 저장하는 연산자

배열의 크기가 다를 경우, 크기가 큰 배열을 기준으로 동작하고,

작은 배열의 존재하지 않는 위치의 원소값은 0으로 처리함.

IntArray& operator+=(const IntArray& rhs);

[] 연산자: 배열의 [] 연산자와 동일한 기능을 수행하는 연산자

int& operator[](int idx);

Implicit inline 함수로 구현

5. public 메소드

int GetSize() : 배열의 크기를 구하는 함수

Implicit inline 함수로 구현

void Add(int idx, int val) : 배열의 *idx*위치에 정수 *val*을 추가하는 함수

반환값 : 정수 *val*이 추가된 배열의 위치(index)

int Remove(int val) : 배열에서 val 값에 해당하는 모든 값을 삭제하는 함수

반환값 : 배열에서 삭제된 정수의 개수

void RemoveAll() : 크기가 0인 빈 배열로 만드는 함수

※ 배열의 인덱스를 매개변수로 갖는 함수([]연산자, Add())에서 입력된 *idx*는 배열 범위(0 ~ m_nSize-1)를 벗어나지 않는다고 가정한다. 즉, 함수 내에서 *idx*의 범위를 검사하는 코드는 추가하지 않아도 된다.

※ 배열의 크기가 변경되는 경우 크기에 맞게 메모리를 동적으로 관리해야 한다.

- 다음의 코드를 이용하여 생성된 IntArray를 테스트하시오.

```
int i;
IntArray a1;

// 0-99 사이의 수를 무작위로 5개 발생하여 a1의 Add를 이용하여 배열의 맨 앞에 추가
// rand 함수 이용할 것.


// 10, 8, 6, 4, 2 값을 a1의 맨 앞에 차례로 추가

cout << "a1 : " << a1 << endl;      // a1 화면 출력

IntArray a2(a1);     // a2 : a1으로부터 복사 생성
cout << "a2 : " << a2 << endl;      // a2 화면 출력

int nDel;
nDel = a1.Remove(0);                    // a1에서 0값 모두 삭제
cout << nDel << "개 삭제." << endl;     // 삭제된 개수 출력
nDel = a1.Remove(8);                    // a1에서 8값 모두 삭제
cout << nDel << "개 삭제." << endl;     // 삭제된 개수 출력

cout << "a1 : " << a1 << endl;  // 삭제 후 a1 화면 출력

IntArray a3(5);      // a3 : 크기가 5인 배열 생성
for (i = 0; i < 5; i++)
    a3[i] = i;       // [] 연산을 이용하여 0-4의 수로 초기화

cout << "a3 : " << a3 << endl;  // a3 화면 출력

a2 = a3 + a1;        // 배열의 + 연산
cout << "a2(a3 + a1) : " << a2 << endl; // + 연산 후의 a2 화면 출력

a3 += a1;            // 배열의 += 연산

cout << "a3(a3 += a1) : ";              // += 연산 후의 a3 화면 출력
for (i = 0; i < a3.GetSize(); i++)
    cout << a3[i] << " ";
cout << endl;
```

출력 결과 예

```
C:₩Windows₩system32₩cmd.exe
a1 : 2 4 6 8 10 38 87 48 31 50
a2 : 2 4 6 8 10 38 87 48 31 50
0개 삭제.
1개 삭제.
a1 : 2 4 6 10 38 87 48 31 50
a3 : 0 1 2 3 4
a2(a3 + a1) : 0 1 2 3 4 2 4 6 10 38 87 48 31 50
a3(a3 += a1) : 2 5 8 13 42 87 48 31 50
계속하려면 아무 키나 누르십시오 . . .
```

경희대학교 KYUNG HEE UNIVERSITY

다음의 조건을 만족하는 2차원 구조체 배열을 생성하시오.

1. 구조체를 struct를 이용하여 다음과 같이 정의

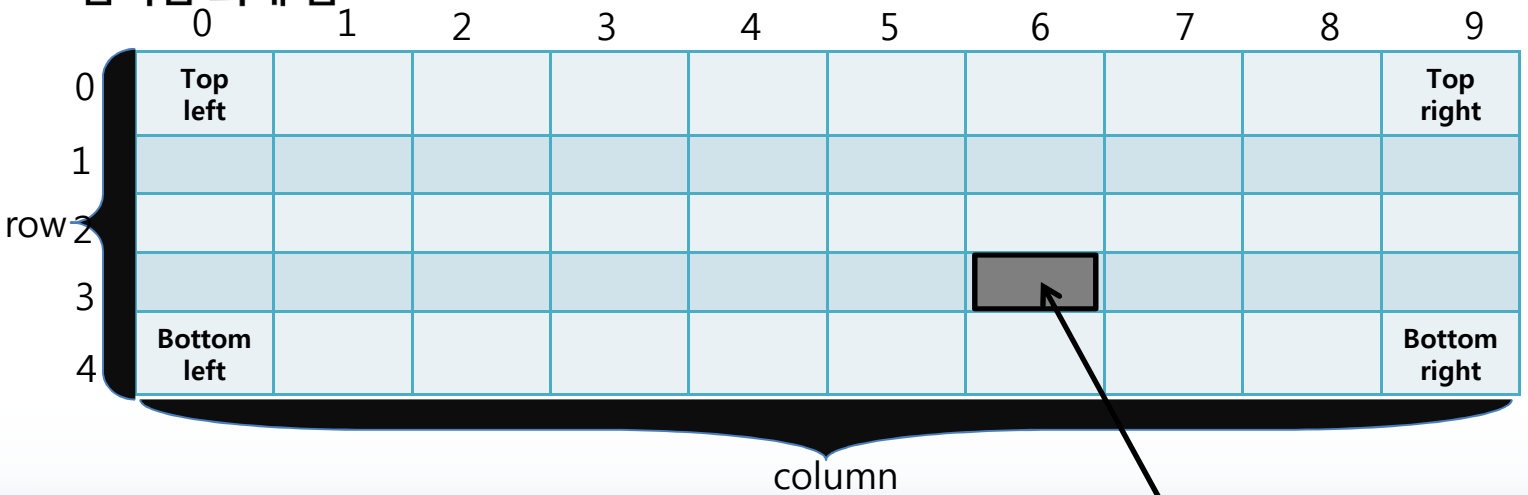
   구조체의 이름 : mapitem

   구조체의 멤버 변수 :

   int 형 : row, column (2차원 배열표현시 row, column 정보를 가짐)

   char* 형 : line (해당 위치의 문자열 정보)

2. 2차원 그리드를 동적 할당으로 구현하는 제시된 코드를 이용하여 논리적으로 다음 그림처럼 되게 함.



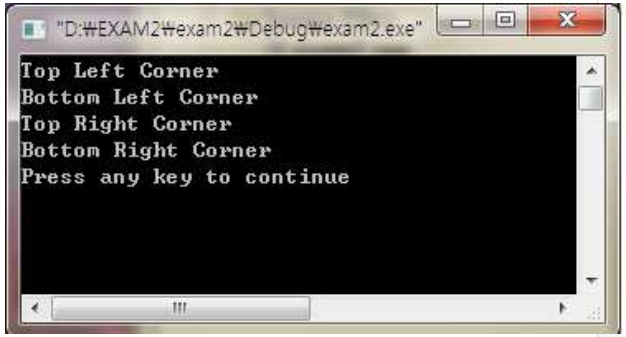
3. 다음 지정된 위치의 정보(line 멤버 변수)를 출력

(0,0), (0,9),(4,0),(4,9)

다음의 코드를 이용하여 주어진 문제를 해결하시오.

```
#include <cstdlib>
#include <iostream>
using namespace std;
#define MAXROWS 5
#define MAXCOLUMNS 10
#define MAXLINELENGTH 80

//mapitem구조체 정의

typedef ;              // mapitem*를 pmapitem으로 타입 정의
typedef pmapitem row[MAXCOLUMNS];
typedef  ;                    // row*의 형을 갖는 MAXROWS의 크기로 gridtype을 선언
int main(int argc, char* argv[])
 {
        gridtype grid;
        pmapitem tempitem;
        int rowindex,colindex;
        for (rowindex=0;rowindex < MAXROWS; rowindex++) {
                grid[rowindex]=  ;                                  // row 하나를 동적 할당함
                for (colindex=0;colindex < MAXCOLUMNS;colindex++){
                        tempitem = ;                                // tempitem에 적절한 값을 할당
                        tempitem->row= rowindex;
                        tempitem->column= colindex;

                        // tempitem의 line에 MAXLINELENGTH 길이의 Char 형을 동적 할당 및 null string(길이 0
                           인 문자열으로 초기화.
                       //  grid의 ( rowindex. colindex )위치에 tempitem 값 할당.
                 }
        }

    strcpy((*grid[0][0])->line, "Top Left Corner");
    strcpy((*grid[MAXROWS-1][0])->line, "Bottom Left Corner");
    strcpy((*grid[0][MAXCOLUMNS-1])->line="Top Right Corner");
    strcpy((*grid[MAXROWS-1][MAXCOLUMNS-1])->line, "Bottom Right Corner"(;

    /* 출력 코드 작성 */

    return 0;
}
```

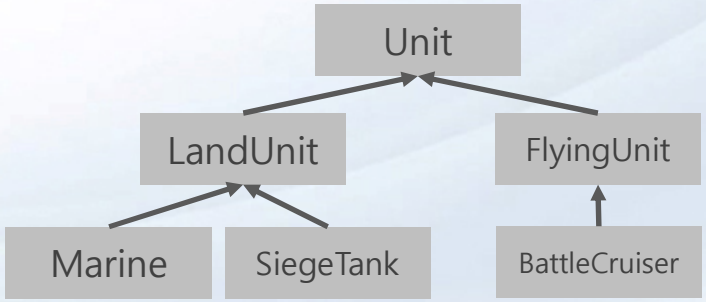
출력 결과 예



스타크래프트의 유닛 관리를 위한 클래스들을 구현하라.

1. 아래와 같은 클래스 상속 구조를 가지는 클래스들을 구현하라.
   1. Unit 은 최상위 클래스로 모든 유닛이 공통적으로 가지는 정보를 멤버 변수로, 공통적인 함수를 가상 (virtual) 메소드로 가짐.
   2. LandUnit 은 육상유닛들의 상위 클래스로 유닛의 2차원 위치를, FlyingUnit은 공중유닛들의 상위 클래스로 유닛의 3차원 위치를 멤버변수로 가짐.
   3. 최하위 클래스들은 각각의 유닛이 가지고 있는 special skill을 관리하는 변수를 가짐.
2. 각 클래스의 정확한 변수, 메소드 리스트는 다음장의 표 참조
3. 각각의 클래스들의 생성자를 적절히 만들고, 초기화 리스트 방법을 사용하여 마지막 페이지의 출력 결과를 참고하여 멤버 변수들의 값을 초기화 해야 함.
4. 메인함수는 다다음장의 코드를 따름. 메인함수에서는 최상위 클래스인 Unit 클래스의 포인터 변수만 선언하고 최하위 클래스들을 dynamic binding 하여 사용할 수 있게 클래스들을 구성해야 함.

상속구조



클래스 정보

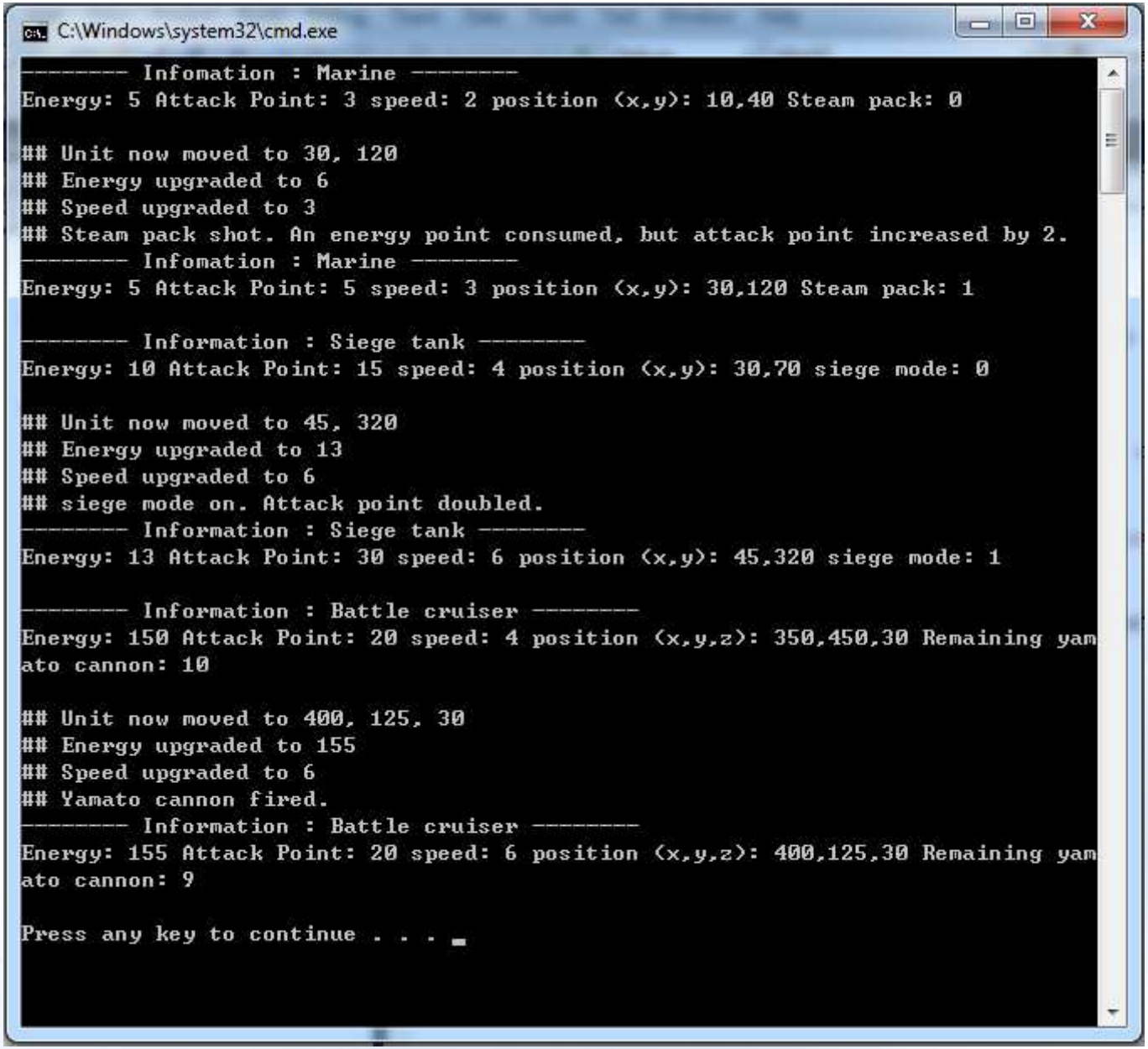
경희대학교 KYUNG HEE UNIVERSITY

Unit	int energy: 유닛의 에너지 int attackPoint: 유닛의 공격력 int speed: 유닛의 속도 void toggleSpecialSkill(): 속이 빈 가상함수 void speedUpgrade(): 가상함수. speed 변수의 값을 1 증가시키고 결과를 출력 void energyUpgrade(): 가상함수. energy변수의 값을 1 증가시키고 결과를 출력 void printUnitInfo(): 가상함수. 위 3가지 변수의 값을 출력 void moveTo(int*): 속이 빈 가상함수
LandUnit	int position[2]: 유닛의 2차원 위치를 관리하는 변수 void moveTo(int*): 2차원 array를 받아 position변수에 카피. Unit클래스의 동명 함수를 재 정의 void printUnitInfo(): position을 출력하는 함수
FlyingUnit	int position[3]: 유닛의 3차원 위치를 관리하는 변수 void moveTo(int*): 3차원 array를 받아 position변수에 카피. Unit클래스의 동명 함수를 재 정의 void printUnitInfo(): position을 출력하는 함수
Marine	bool isSteamPack: 스팀팩 special skill이 사용되는지 아닌지 저장하는 변수 void toggleSpecialSkill(): Unit의 동명함수를 재정의. isSteamPack변수를 on 혹은off로 스위치. Off에서 on으로 바뀔때는 에너지 1감소하고 attackPoint2 증가. 반대일 경우는 attackPoint 2 감소. 이 변화를 화면에 출력. void printUnitInfo(): 상위 동명함수를 사용하여 유닛의 모든 정보를 출력.
SiegeTank	bool isSiegeMode: 탱크가 시즈모드인지 아닌지 저장하는 변수 void toggleSpecialSkill(): Unit의 동명함수를 재정의. isSiegeMode변수를 on 혹은off로 스위치. Off에서 on으로 바뀔때는 attackPoint가 배로 증가. 반대일 경우는 attackPoint가 반으로 감소. 이 변화를 화면에 출력. void printUnitInfo(): 상위 동명함수를 사용하여 유닛의 모든 정보를 출력. void speedUpgrade(): Unit클래스의 동명함수를 overriding하여 speed값을 1이 아닌 2 증가시킴. void energyUpgrade(): Unit클래스의 동명함수를 overriding하여 energy값을 1이 아닌 3 증가시킴.
BattleCruiser	int numYamatoCannon: 아마토포가 얼마나 남아있는지 저장하는 변수 void toggleSpecialSkill(): Unit의 동명함수를 재정의. 아마토포를 발사. numYamatoCannon을 1 감소시킴. 이 변화를 화면에 출력. void printUnitInfo(): 상위 동명함수를 사용하여 유닛의 모든 정보를 출력. void speedUpgrade(): Unit클래스의 동명함수를 overriding하여 speed값을 1이 아닌 2 증가시킴. void energyUpgrade(): Unit클래스의 동명함수를 overriding하여 energy값을 1이 아닌 5 증가시킴.

Main.cpp 파일

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

int main() {
    Unit *myUnit[3];
    int position[3];

    //create 3 different units
    myUnit[0] = new Marine();
    myUnit[1] = new SiegeTank();
    myUnit[2] = new BattleCruiser();

    for(int u = 0; u < 3; u++) {
        myUnit[u]->printUnitInfo();
        if(u == 0) {
            position[0] = 30; position[1] = 120;
        } else if(u == 1) {
            position[0] = 45; position[1] = 320;
        } else if(u == 2) {
            position[0] = 400; position[1] = 125; position[2] = 30;
        }
        myUnit[u]->moveTo(position);
        myUnit[u]->energyUpgrade();
        myUnit[u]->speedUpgrade();
        myUnit[u]->toggleSpecialSkill();
        myUnit[u]->printUnitInfo();
    }
    return 0;
}
```

출력 결과


C:\Windows\system32\cmd.exe
-------- Infomation : Marine --------
Energy: 5 Attack Point: 3 speed: 2 position (x,y): 10,40 Steam pack: 0
Unit now moved to 30, 120
Energy upgraded to 6
Speed upgraded to 3
Steam pack shot. An energy point consumed, but attack point increased by 2.
-------- Infomation : Marine --------
Energy: 5 Attack Point: 5 speed: 3 position (x,y): 30,120 Steam pack: 1
-------- Information : Siege tank --------
Energy: 10 Attack Point: 15 speed: 4 position (x,y): 30,70 siege mode: 0
Unit now moved to 45, 320
Energy upgraded to 13
Speed upgraded to 6
siege mode on. Attack point doubled.
-------- Information : Siege tank --------
Energy: 13 Attack Point: 30 speed: 6 position (x,y): 45,320 siege mode: 1
-------- Information : Battle cruiser --------
Energy: 150 Attack Point: 20 speed: 4 position (x,y,z): 350,450,30 Remaining yam
ato cannon: 10
Unit now moved to 400, 125, 30
Energy upgraded to 155
Speed upgraded to 6
Yamato cannon fired.
-------- Information : Battle cruiser --------
Energy: 155 Attack Point: 20 speed: 6 position (x,y,z): 400,125,30 Remaining yam
ato cannon: 9
Press any key to continue . . .

대형마트를 위한 영수증 관리 프로그램을 구현하라.

목표: K마트에서 사용할 구매 물건에 대한 총액 계산과 포인트 유지를 위한 프로그램을 구현한다.

1. 오늘 K마트에서 판매한 시장바구니(market basket)들의 리스트가 주어졌을 때, 시장바구니 별로 구매 총액을 계산하고 그 바구니에 해당하는 고객의 포인트 추가 점수를 계산한다. 또한, 오늘 총 판매액과 오늘 시장바구니에 나타난 고객 및 제품에 한해, 고객별 포인트, 제품 별 판매 수량을 출력한다.
2. 포인트 제도: 결제 방법에 따라 시장바구니 구매액의 일정 비율을 포인트로 적립.
3. 입력은 K마트에서 판매하는 제품목록, K마트 고객목록, 포인트 적립비율목록, 시장바구니 목록이 순서대로 구성된다. 각 목록 사이에는 하나의 빈 줄이 있음.
   - 제품목록: 첫 줄은 제품 개수. 이어서 각 줄에 제품 하나에 대한 제품이름, 가격의 쌍으로 된 정보가 있음. 제품이름은 최대 20자. 최대 제품 수는 100개.
   - 고객목록: 첫 줄은 고객 수. 이어서 한 줄에 한 고객의 정보가 나옴. 고객정보는 고객번호, 고객이름, 포인트로 구성됨. 고객이름은 최대 10자.
   - 포인트 적립비율목록: 첫 줄은 결제방식 개수. 한 줄에 하나씩 결제방법 별 포인트 적립비율이 나옴. 결제방법은 최대 20자임. 적립비율은 단위가 % 임.
   - 시장바구니 목록: 각 시장바구니는 첫 줄에 시장바구니번호, 고객번호, 결제방법, 구매제품 종류 수가 나옴. 이어서 각 줄에 구매제품 별로, 제품이름과 구매 개수가 나옴. 시장바구니번호가 0이면 시장바구니 목록의 끝을 나타냄.

대형마트를 위한 영수증 관리 프로그램을 구현하라.

4. 출력은 다음 페이지 출력 예처럼 바구니 판매액, 고객 포인트 변동 내역, 제품별 판매량을 출력한다.
5. 주어진 main.cpp 는 프로그램의 일부분이다. 빠진 부분을 채워서 프로그램을 완성하시오. 주어진 프로그램 코드는 변경하면 안 됨.
   - 모든 동적 메모리 할당은 main() 함수가 아닌 호출된 함수에서 수행해야 함.
   - 메모리 반환은 main() 함수 내에서 수행

참고: 문자열 다루는 예

```
#include <cstring>
…
      char str[] = "Superman";

      int len = strlen(str);
      char *aCopy = new char[len+1];
      strcpy(aCopy, str);
      if (strcmp(aCopy, str) == 0)
            cout << “same” << endl;
      else
            cout << “different” << endl;
```

Debugging Tip: standard input from file when debugging from Visual Studio

- This explains how to use standard input redirection in Visual Studio.
- Right-click on the project in the Solution Explorer and select “Open Folder in Windows Explorer”. Create a text file that contains the input. Suppose the filename is input.txt.
- Right-click on the project in the Solution Explorer and go to Properties. Then click on "Debugging" on the left and in the "Command Arguments" box type something like < input.txt

Exam 2 | **Question 4**

- 입력 예

```
5
milk 500
bread 1000
butter 2000
coke 700
chocolate 1500

3
3798 John 751
4801 Smith 10150
8359 Jane 3786

2
cash 20
credit 10

350 3798 cash 2
milk 3
coke 2
351 8359 credit 1
bread 2
0
```

- 출력 예

```
### 바구니 판매액 ###
바구니 350: (고객명: John, 총액: 2900, 포인트: 580)
바구니 351: (고객명: Jane, 총액: 2000, 포인트: 200)
총액: 4900

### 고객 포인트 변동 내역 ###
3798 John 1331
8359 Jane 3986

### 제품별 판매량 ###
milk 3
coke 2
bread 2
```

Main.cpp 파일

```
struct Product {
    // fill up the members
    // use static array for the product name
};

struct Customer {
    // fill up the members
    // use a pointer for the customer name
};

const int MAX_PRODUCTS=100;

Product* findProduct(Product *products[], char *name);
Customer* findCustomer(Customer customers[], int nCustomers, int cust_id);
Payment* findPayment(Payment payments[], int nPayments, char *name);

int main( )
{

    Product *products[MAX_PRODUCTS+1]; // array of pointers, each of which points to a product
    Customer *customers; // array of customers
    Payment *payments; // array of payments
    Basket aBasket;
    int nCustomers, nPayments;

    getProductList(products);
    getCustomerList(&customers, &nCustomers);
    getPointPaymentList(&payments, nPayments);

    while (getBasket(aBasket))
    {
    }

    return 0;
}
```