Exam #4

경희대학교
KYUNG HEE UNIVERSITY

- **정수 집합 S를 입력 받아, S 안에 존재하는 서로 다른 원소 a, b, c, d에 대하여 a+b+c=d를 만족하는 가장 큰 숫자 d를 찾는 프로그램을 작성하시오.**

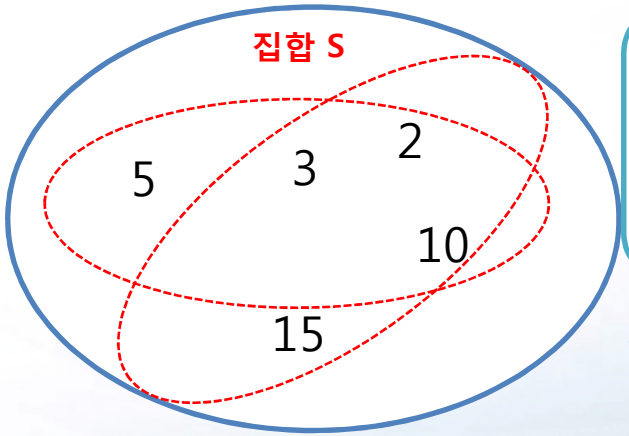
정수 집합 S를 입력 받아, S 안에 존재하는 서로 다른 원소 a, b, c, d에 대하여 a+b+c=d를 만족하는 가장 큰 숫자 d를 찾는 프로그램을 작성하시오.

<입력 형식 및 조건>

1. 입력의 첫 번째 줄에는 집합 S안의 원소의 개수 n(1≤n≤1,000)이 입력된다.
2. 그 다음 줄부터 n개의 줄에는 집합 S의 원소를 입력한다.
3. 집합 내에 중복되는 원소는 존재하지 않으며 원소의 범위는 -536,870,912 ~ +536,870,911 사이임을 가정한다.

<출력 형식>

1. 집합 S에 대해 위의 조건을 만족하는 가장 큰 d를 출력한다.
2. 만약 a+b+c=d를 만족하는 경우가 집합 S안에 존재 하지 않을 경우 "no solution"을 출력한다.



[집합 S에 따른 출력 예시]

- 5+3+2=10
- 3+2+10=15
이므로, a+b+c=d를 만족하는 가장 큰 d인 15를 출력

```
5
5
10
3
2
15
Max value : 15
```

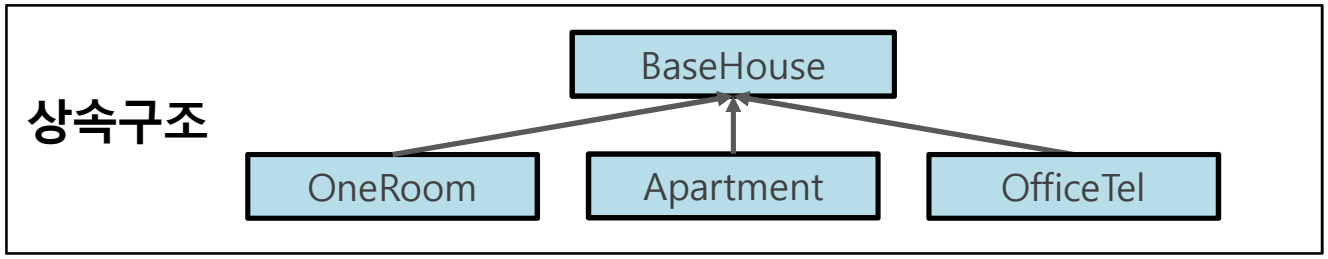
[실행의 예]
- 조건을 만족하는 d가 존재하는 경우

```
5
2
16
64
256
1024
No Solution!
```

[실행의 예]
- 조건을 만족하는 d가 존재하지 않는 경우

주거용 부동산 관리를 위한 클래스들을 구현하라.

1. 주거용 부동산은 원룸, 아파트, 오피스텔 세 종류의 주거 형태를 나타낸다. 각 주거 형태는 다음과 같은 클래스 상속 구조를 가지고 있다.



2. BaseHouse 클래스는 순수 가상 함수를(Pure Virtual Function) 포함 하고 있는 추상 클래스(Abstract Class) 이다.
3. OneRoom, Apartment, OfficeTel 클래스는 BaseHouse의 자식 클래스(public 상속)들 이고 이들 클래스는 BaseHouse에서 선언된 순수 가상 함수를(Pure Virtual Function) 구현(Definition) 객체화 가능하도록 만든다.
4. 각 클래스의 정확한 변수, 메소드 리스트는 다음장의 표 참조
5. 단, 다음장의 표는 각 클래스의 소멸자(Destructor)가 일반 소멸자(Destructor) 인지 가상 소멸자(Destructor)에 기술되어 있지 않다. 적절하게 가상 소멸자와 일반 소멸 자를 선택 각 클래스를 선언한다.
6. 메인함수는 뒤에 나오는 장의 코드를 따름.

클래스 정보

경희대학교 KYUNG HEE UNIVERSITY

BaseHouse	멤버 변수(protected)

	char *mAddress; // 집 주소 (최대크기 200자로 동적 할당)
	int mSize; // 집 크기 (m제곱 단위)
	int mConstructYear; // 집 완공 년도
	int mFloors; // 집 층수 (-는 지하를 의미)
	int mPrice; // 집 가격 (단위 원)
	int mRentPrice; // 월세 가격 (단위 원)
	int mRepository; // 월세 보증금 (단위 원)

	멤버 함수(public)

	virtual void ShowHouseInfo() =0; // 집 정보 출력하는 순수 가상함수
	BaseHouse(); // 생성자 mAddress 동적할당
	~BaseHouse(); // 소멸자 mAddress 동적 할당 해제
	void SetAddress(char *addr); // 주소를 입력 받는 함수 strcpy를 사용해서 문자열 복사
	void SetSize(int size); // 집 크기를 입력 받는 함수
	void SetContructYear(int year); // 집 완공 년도를 입력 받는 함수
	void SetFloors(int floors); // 집 층수를 입력 받는 함수
	void SetPrice(int price); // 집 매매 가격을 입력 받는 함수
	void SetRentPrice(int rentprice); // 집 월세 가격을 입력 받는 함수
	void SetRepository(int repository);// 집 월세 보증금을 입력 받는 함수
	void PrintBasicInfo(); // 집 주소, 집 크기, 완공 년도, 집 층수, 매매 가격, 월세가격, 월세 보증금을 다다음 페이지 출력 결과와 같이 화면에 출력하는 함수

OneRoom	멤버 변수(protected)

	bool mBed; // 침대 기본 제공, true 침대 제공, false 제공 안함
	bool mCleaner; // 세탁기 제공 여부, true 세탁기 제공, false 제공 안함
	bool mElectronicCharge; // 전기세 별도 부과 여부, true 별도 부과, false 별도 부과 안함
	bool mWaterCharge; // 수도세 별도 부과 여부, true 별도 부과, false 별도 부과 안함

	멤버 함수(public)

	virtual void ShowHouseInfo(); // 원룸 정보 출력 함수
	void SetOneRoomOptions(bool bed, bool cleaner, bool ec, bool wc); // 침대, 세탁기, 전기세, 수도세 관련 옵션 설정
	OneRoom(); // 생성자, 옵션을 나타내는 멤버 변수를 모두 false로 기본 설정 해준다.
	~OneRoom(); // 소멸자

클래스 정보2

경희대학교 KYUNG HEE UNIVERSITY

Apartment	멤버 변수(protected)

	int mNumRoom; // 방 개수
	int mNumToilet; // 화장실 개수
	bool mDistrictHeating; // 지역 난방 유무, true 지역난방, false 지역난방 아님

	멤버 함수(public)

	virtual void ShowHouseInfo(); // 아파트 정보 출력 함수
	Apartment(); // 생성자, 방 개수, 화장실 개수를 0으로 초기화 지역 난방 유무는 true로 초기화
	~Apartment(); // 소멸자
	void SetAptOptions(int nRoom, int nToilet, bool dh); // 방 개수, 화장실 개수, 지역난방 유무 설정

OfficeTel	멤버 변수(protected)

	bool mDuplexFloor; // 오피스텔 복층 유무, true 복층, false 단층
	bool mParking; // 주차장 제공 여부, true 주차장 제공, false 주차장 미제공
	bool mIndustrialElectric; // 산업용 전기 사용 가능 여부, true 산업용 전기 사용가능, false 불가능
	bool mTv; // TV기본 제공 여부, true TV 제공, false TV 미제공

	멤버 함수(public)

	virtual void ShowHouseInfo(); // 오피스텔 정보 출력 함수
	OfficeTel(); // 생성자
	~OfficeTel (); // 소멸자
	void SetOfficeTelOptions(bool duplexfloor, bool parking, bool ie, bool tv); // 복층, 주자장, 산업용 전기, TV 설정 함수

Main.cpp 파일

```
void main( )
{
    OneRoom *oneroom = new OneRoom;
    Apartment *apt =  new Apartment;
    OfficeTel *officetel = new OfficeTel;

    oneroom->SetAddress("경기도 수원시 영통구 영통동 100x-xx");
    oneroom->SetSize(20);
    oneroom->SetContructYear(1998);
    oneroom->SetFloors(-1);
    oneroom->SetPrice(100000000);
    oneroom->SetRentPrice(250000);
    oneroom->SetRepository(3000000);
    oneroom->SetOneRoomOptions(true,false,false,true);

    apt->SetAddress("경기도 수원시 영통구 영통동 신나무실 아파트 oo동 oo호");
    apt->SetSize(84);
    apt->SetContructYear(2001);
    apt->SetFloors(7);
    apt->SetPrice(340000000);
    apt->SetRentPrice(650000);
    apt->SetRepository(20000000);
    apt->SetAptOptions(4,2,true);

    officetel->SetAddress("경기도 수원시 영통구 영통동 유니빌 ooo호");
    officetel->SetSize(46);
    officetel->SetContructYear(2004);
    officetel->SetFloors(10);
    officetel->SetPrice(150000000);
    officetel->SetRentPrice(500000);
    officetel->SetRepository(10000000);
    officetel->SetOfficeTelOptions(false,false,true,true);

    oneroom->ShowHouseInfo( );
    cout<<endl;
    apt->ShowHouseInfo( );
    cout<<endl;
    officetel->ShowHouseInfo( );

}
```

출력 결과


```
원룸 정보
주소 : 경기도 수원시 영통구 영통동 100x-xx
크기 : 20   완공년도: 1998
지하 1 층   매매가 : 100000000
월세가격 : 250000 원      월세 보증금: 3000000 원
침대 기본 제공 : 0      세탁기 기본 제공 : X
전기세 별도 부과 : X      수도세 별도 부과 : 0

아파트 정보
주소 : 경기도 수원시 영통구 영통동 신나무실 아파트 oo동 oo호
크기 : 84   완공년도: 2001
지상 7 층   매매가 : 340000000
월세가격 : 650000 원      월세 보증금: 20000000 원
방 개수  : 4    화장실 개수 : 2
지역 난방 : 0

오피스텔 정보
주소 : 경기도 수원시 영통구 영통동 유니빌 ooo호
크기 : 46   완공년도: 2004
지상 10 층   매매가 : 150000000
월세가격 : 500000 원      월세 보증금: 10000000 원
복층 : X      주차장 : X
산업용 전기 : 0      TV 기본 제공 : 0
계속하려면 아무 키나 누르십시오 . . .
```

템플릿을 이용한 스마트 포인터 클래스 구현

C++는 동적 할당을 이용하면 반드시 동적할당을 해제 해줘야 한다. 하지만 동적 할당을 해제 하지 않아서 생기는 메모리 누수(Memory Leak)현상이 빈번하게 발생한다. 이를 줄이고 사용자의 편의성으로 높이기 위한 SmartPointer 클래스를 작성한다.

다음 페이지의 SmartPointer 와 ReferenceCount 클래스 선언을 기반으로 클래스를 완성하고 제시한 main함수 코드를 이용해서 테스트 한다.

구현 시 아래 각각의 세부 항목을 만족시킨다. (각 세부 항목당 부분 점수가 있음)

1. SmartPointer 클래스 안에 ReferenceCount 클래스 포인터를 포함하도록 선언
2. 다음 페이지의 SmartPointer와 ReferenceCount 클래스 선언에 빠진 부분을 주석을 참고해서 채워 넣는다.
3. SmartPointer 선언에 존재하는 생성자 3가지를 주석을 참고하여 올바르게 정의한다.
4. SmartPointer 선언에 존재하는 주석을 참고하여 올바르게 소멸자를 정의한다.
5. SmartPointer 선언에 존재하는 *연산자와 -> 연산자를 주석을 참고하여 올바르게 정의한다.
6. SmartPointer 선언에 존재하는 = 연산자를 주석을 참고 올바르게 정의한다.

SmartPointer 클래스 선언

```
template <class Type>  // 템플릿 클래스 선언
class SmartPointer
{
private:
    Type*           mDataPointer;     // 스마트 포인터에 동적할당될 포인터
    ReferenceCount* mReferenceCount;  // 참조 숫자를 세는 ReferenceCount 클래스 포인터

public:
    // 기본 생성자,
    // mDataPointer에 0값 할당, mReferenceCount 동적할당 후, 참조 숫자 1증가(AddRef() 함수 사용)
    SmartPointer();

    // 동적할당된 포인터 값을 받는 생성자
    // mDataPointer에 pValue 값을 넣어주고, mReferenceCount 동적할당 후, 참조 숫자 1증가(AddRef() 함수 사용)
    SmartPointer(Type* pValue);

    // 다른 스마트 포인터 값을 받는 생성자
    // mDataPointer에 sp.mDataPointer 값을 넣어주고, mReferenceCount에 sp.mReferenceCount 값을 넣어준 후
    // 참조 숫자 1증가(AddRef() 함수 사용)
    SmartPointer(const SmartPointer<Type>& sp) : mDataPointer(sp.mDataPointer), mReferenceCount(sp.mReferenceCount);

    // 소멸자
    // mReferenceCount의 Release()함수를 호출 한후 리턴값이 0이면
    // mDataPointer와 mReferenceCount를 동적할당 해제 해준다.
    ~SmartPointer();

    // *연산자
    // mDataPointer 주소에 저장된 값을 리턴
    Type& operator* ();

    // -> 연산자
    // mDataPointer 값을 리턴
    Type* operator-> ();

    // = 대입 연산자
    // sp의 주소값과 this가 같은지 확인하고 같지 않다면
    // mReferenceCount의 Release를 호출한다. 이때 Release()함수의 리턴 값이 0이면
    // mDataPointer와 mReferenceCount를 동적할당 해제 해준다.
    // sp의 mDataPointer와 mReferenceCount를 this의 mDataPointer와 mReferenceCount에
    // 넣어주고 mReferenceCount의 참조 숫자 1증가(AddRef() 함수 사용)
    // *this 리턴
    SmartPointer<Type>& operator = (const SmartPointer<Type>& sp);
};
```

ReferenceCount 클래스 선언 및 정의

```
class ReferenceCount
{
private:
    int mRCount; // Reference count
public:
    void AddRef(){ // 참조 숫자 증가
        mRCount++;
    }
    int Release(){ // 참조 숫자 감소 후 감소된 값 리턴
        return --mRCount;
    }
    ReferenceCount(){mRCount=0;}
};
```

Main.cpp 파일

```
using namespace std;

class C1
{
public:
    int mCValue;
    ~C1(){cout<<"Delete C1"<<endl;}
};

int main()
{
    SmartPointer<C1> spC1_1;         // 스마트 포인터 생성
    SmartPointer<C1> spC1_2(new C1);// C1 동적할당 값을 spC1_2에 넣어준다.
    SmartPointer<C1> spC1_3(spC1_2);// spC1_2 값을 spC1_3에 복사

    spC1_1 = spC1_3;                 // spC1_1에도 spC1_3값 복사

    // spC1_1, spC1_2, spC1_3 모두 같은 포인터 값을 가짐
    spC1_1->mCValue = 100;
    cout<<"spC1_1->mCValue : "<<spC1_1->mCValue<<endl;
    cout<<"spC1_2->mCValue : "<<(*spC1_2).mCValue<<endl;
    cout<<"spC1_3->mCValue : "<<spC1_3->mCValue<<endl;

    spC1_1 =0;
    spC1_2 =0;
    cout<<"주소값을 가지는 객체가 없을때 동적할당된 값 자동 해제"<<endl;
    spC1_3 =0;

    SmartPointer<C1> spC1_4(new C1);// C1 동적할당 값을 spC1_4에 넣어준다.

    // 프로그램이 끝날때 자동 동적 할당 해제
    cout<<"프로그램이 끝나면서 자동 동적할당 해제"<<endl;

    return 0;
}
```

실행 화면

```
spC1_1->mCValue : 100
spC1_2->mCValue : 100
spC1_3->mCValue : 100
주소값을 가지는 객체가 없을때 동적할당된 값 자동 해제
Delete C1
프로그램이 끝나면서 자동 동적할당 해제
Delete C1
계속하려면 아무 키나 누르십시오 . . .
```

임의의 문장을 입력 받아 단어 별로 나눈 후, 각 단어들이 나타난 개수를 구하는 프로그램을 작성하시오.

사용자로부터 문자열을 입력 받아, 단어 별로 나누고 각 단어들이 나타난 횟수를 구하여 출력하는 프로그램을 작성하시오. 단, 다음 페이지에 제시된 3개의 함수(SplitSentence, MakeUniqueArray, CountWord)를 구현하여 프로그램을 작성한다.

1. 사용자로부터 200자 이하의 문자열을 입력 받는다.
2. 입력 받은 문자열에서 각 단어가 나타난 횟수를 센다.
3. 입력 받은 문자열에서 나타난 모든 단어들을 아래 [실행의 예]와 같이 출력한다.

<처리 조건>

1. 입력된 문자열에서 각 단어 사이의 구분은 공백으로 한다.
2. 사용자는 문자열로 200자 이내의 알파벳, 공백, 각종 기호들(., !, @, # 등...)을 입력할 것을 가정한다.
3. 각 문장에서 나타나는 단어의 수는 100개 이하임을 가정한다.
4. 각 단어의 길이는 100 이하임을 가정한다.

```
Your guess is as good as mine.

Your      : 1
guess     : 1
is        : 1
as        : 2
good      : 1
mine.     : 1
```

```
What's the title of the movie? When do you show the movie?

What's    : 1
the       : 3
title     : 1
of        : 1
movie?    : 2
When      : 1
do        : 1
you       : 1
show      : 1
```

[실행의 예]

함수 설명

1. SplitSentence
   - 인자 값 : 분리할 문자열, 단어 배열
   - 반환 값 : 분리된 단어들의 개수
   - 기능 : 문자열을 공백 단위로 분리하여 단어 배열에 저장
2. MakeUniqueArray
   - 인자 값 : 중복을 제거한 단어 배열, 단어 배열, 중복 제거 전의 단어 배열의 크기
   - 반환 값 : 중복을 제거한 단어 배열의 크기
   - 기능 : 단어 배열로 부터 중복을 제거하고, 중복을 제거한 단어 배열에 저장
3. CountWord
   - 인자 값 : 단어 배열, 중복 제거 단어 배열, 단어 배열의 크기, 중복 제거 단어 배열의 크기, 각 단어의 개수를 저장하기 위한 정수형 배열
   - 반환 값 : 없음
   - 기능 : 각 단어가 나타난 횟수를 세어, 해당 단어가 나타난 횟수를 중복 제거 단어 배열에서 단어가 나타난 인덱스 번지의 정수형 배열에 저장.

[참고] strtok 함수 (문자열 분리 함수)

1. 함수 프로토타입

   char * strtok (char * str, const char * delimiters);

   str은 문자열, delimiters는 문자열을 분리할 때의 구분자

2. 함수 사용 예제

```
char str[] ="- This, a sample string.";
char * pch;
printf ("Splitting string \"%s\" into tokens:\n",str);
pch = strtok (str," ,.-");
while (pch != NULL)
{
  printf ("%s\n",pch);
  pch = strtok (NULL, " ,.-");
}
```

[출력의 예]

```
Splitting string "- This, a sample string." into tokens:
This
a
sample
string
```

예외처리 프로그램을 작성하시오.

전체집합이 1부터 100사이의 정수라고 가정하자. 두 개의 집합을 입력받아 두 집합의 교집합을 구하는 프로그램을 아래와 같이 작성하시오.

- 입력은 두 줄로 구성되며, 각 줄은 하나의 집합에 속하는 여러 개의 정수들로 구성된다. 첫째 줄은 집합 A, 둘째 줄은 집합 B를 나타낸다.
- 두 집합 A, B를 입력으로 받아 교집합을 구하는 함수 getIntersection(int setA[], int setB[], int setR[])을 작성한다. setR은 교집합 결과를 리턴한다. 배열에 0의 값을 갖는 element가 집합의 끝을 나타내도록 한다.
- getIntersection() 함수는 아래 두개의 exception을 발생시킬 수 있도록 하고 함수에 exception specification도 부여한다.
  - BigNumber: 100보다 큰 값이 있음. 먼저 발견된 big number 하나에 대해 집합 이름과 그 집합의 첫 번째 big number를 정보로 보내줘야 한다. 집합 A, B에 big number가 모두 있는 경우 집합 A에 있는 것을 보내 준다.
  - NoCommon: 교집합이 공집합임.
- main()함수는 exception 처리를 하고 교집합 결과를 출력한다.

```
1 3 5 7 103 9
2 4 6 8 11 13
Big Number : 103 in set A
계속하려면 아무 키나 누르십시오 . . .
```

```
1 3 5 7 9 11 13 15 17 19 21 23 25
2 4 6 8 10 12 14 16 18 20 22 24 26
No Common element
```

고급 I/O 기능을 이용한 Health Record 관리 프로그램을 작성하시오.

아래와 같이 선언된 char형 배열을 사용하여 요구하는 프로그램을 작성하시오.

```
char input[] = "John 20 170.5 ₩
               Tom 19 168.3 ₩
               Jane 21 165.7 ₩
               Alice 20 168.1";
```

- 이름, 나이, 키로 구성된 HealthRecord struct를 선언한다. 이름의 최대 길이는 20이다.
- istream 객체로부터 HealthRecord를 하나 읽어들이는 bool get_health_record(istream&, HealthRecord&) 함수를 작성한다. 이 함수는 HealthRecord를 성공적으로 읽으면 true를 실패하면 false를 반환한다.
- ostream 객체에 HealthRecord 하나를 아래 출력과 같은 형태로 기록하는 void put_health_record(ostream&, HealthRecord&) 함수를 작성한다.
- 위에 주어진 배열을 이용하여 istringstream 객체를 만들고 get_health_record()함수를 이용하여 레코드를 하나씩 읽어서 "health.bin" 파일에 binary mode로 저장한다. ostringstream 객체 생성 후 "health.bin" 파일로부터 기록된 레코드들을 읽어서 put_health_record() 함수를 이용하여 ostringstream 객체에 기록 후, ostringstream 객체에 있는 문자열을 이용하여 화면에 출력하시오.

실행 결과

```
John|20|170.5
Tom|19|168.3
Jane|21|165.7
Alice|20|168.1
계속하려면 아무 키나 누르십시오 . . .
```

시험 안내

경희대학교 KYUNG HEE UNIVERSITY

- **주의 사항**
  - 참고할 자료들을 미리 다운로드 받은 후, 시험 시작 5분 전 네트워크 차단
  - 스마트 폰, 전자사전 등의 전자기기 일절 사용 금지 (전원 종료 후 가방에 넣지 않으면 부정행위로 간주)
  - 시험 시작 후 출입 불가
- **업 로드 방법**
  - 실습/과제와 동일한 방법으로 솔루션 구성(LAB Help 자료 참고)
  - http://dke.khu.ac.kr/aoop 의 [Assignments Submission] 게시판에 업 로드
  - 파일명과 게시물 제목을 아래와 같이 통일
    - [분반]_ex04_학번
    - Ex) [A]_ex04_2012345678