Exam #2

경희대학교
KYUNG HEE UNIVERSITY

양의 정수가 주어졌을 때, 역 팩토리얼(inverse factorial)을 계산하고 출력하는 프로그램을 작성하시오.

다음과 같이 사용자로부터 양의 정수를 입력 받아 역 팩토리얼(inverse factorial)(n!에서의 n) 을 계산하여 출력하는 프로그램을 작성한다.

1. 최대 k개의 입력이 허용되며 k는 프로그램이 시작할 때, 사용자의 입력을 통해 결정한다.
2. 1부터 2^{32}-1 사이의 양의 정수를 입력 받는다.
3. InverseFactorial 함수를 호출하여 역 팩토리얼 값을 계산한다.
4. 입력 받은 값이 어떤 수의 팩토리얼 값이라면, inputArray 배열에 입력값을, factArray 배열에 역 팩토리얼 값을 저장한다.
5. 만일 입력 받은 값이 어떤 수의 팩토리얼 값이 아니라면, "팩토리얼의 결과값이 아닙니다."라고 출력하고 현재까지의 입력값과 역 팩토리얼 값을 출력하는 printHistory함수를 호출한 후 프로그램을 종료한다.

※ 팩토리얼이란, 1부터 어떤 양의 정수 n까지의 정수를 모두 곱한 것을 말하며 n!으로 나타낸다. 0!=1, 1!=1, 2!=2 n!=n*(n-1)!로 정의된다.

InverseFactorial 함수

- **기능 :** 1개의 양의 정수를 함수의 인자 값으로 받아, 역 팩토리얼을 계산하여 반환한다.
- **반환값** : 인자값의 역 팩토리얼을 계산하여 반환하되,
  1) 인자값이 1인 경우 1을 반환하며,
  2) 인자 값이 어떤 수의 팩토리얼이 아닌 경우 0을 반환한다.

printHistory 함수

- **기능 :** 2개의 양의 정수 배열(입력값과 역 팩토리얼이 저장된 배열)과 크기를 입력 받아, 배열의 내용을 오른쪽 예와 같이 출력한다.

실행 예 1

```
입력 최대 개수 : 5
양의 정수를 입력하세요. :2
양의 정수를 입력하세요. :479001600
양의 정수를 입력하세요. :1
양의 정수를 입력하세요. :120
양의 정수를 입력하세요. :24

2는 2! 입니다.
479001600는 12! 입니다.
1는 1! 입니다.
120는 5! 입니다.
24는 4! 입니다.
```

k (=5) 개 모두 역 팩토리얼이 존재함

실행 예 2

```
입력 최대 개수 : 5
양의 정수를 입력하세요. :2
양의 정수를 입력하세요. :479001600
양의 정수를 입력하세요. :1
양의 정수를 입력하세요. :120
양의 정수를 입력하세요. :8
8은 팩토리얼의 결과값이 아닙니다.

2는 2! 입니다.
479001600는 12! 입니다.
1는 1! 입니다.
120는 5! 입니다.
```

역 팩토리얼이 존재하지 않는 수가 입력됨

수열을 표현하기 위한 프로그램을 구현하라.

다음 페이지 main.cpp 프로그램의 실행결과가 실행 예와 같이 되도록, 다음 메소드들을 제공하는 클래스 Sequence를 설계하고 구현하시오. 각 메소드의 리턴 타입은 조건에 알맞게 선택한다.

- 생성자 Sequence(int items[], int len): 주어진 items 배열에 있는 항들로 수열을 초기화한다. len는 배열에 있는 원소의 개수를 나타냄.
- 생성자 Sequence(Sequence& other): 주어진 수열 other를 이용하여 객체를 초기화한다.
- LengthIs(): 수열에 있는 항의 개수를 출력한다.
- Concatenate(Sequence &other): 주어진 수열 other의 항들을 원래 수열의 뒤에 추가한다.
- Print(): 수열의 종류를 출력한 후, 수열의 항들을 순서대로 출력한다. 수열의 종류는 등차수열, 등비수열, 기타수열 중 하나이며 아래의 내부 메소드를 사용하여 구현한다.
  - IsArithmetic(): 수열이 등차수열 (Arithmetic Sequence)이면 true를 아니면 false를 리턴한다. 등차수열은 이웃한 항 간의 차이가 항상 같은 수열임.
  - IsGeometric(): 수열이 등비수열 (Geometric Sequence)이면 true를 아니면 false를 리턴한다. 등차수열은 이웃한 항 간의 비가 항상 같은 수열임.

* 수열의 최소 항의 개수는 3, 최대 항의 개수는 100이라고 가정함.
* 수열 항들을 저장하기 위하여 동적 배열을 사용하는 경우 추가 점수 부여.

main.cpp 파일

```
#include <iostream>
#include "Sequence.h"

using namespace std;

int main()
{
    int arithmetic_seq_items[] = {1, 3, 5, 7, 9};
    int geometric_seq_items[] = {1, 2, 4, 8, 16};

    Sequence s1(arithmetic_seq_items, sizeof(arithmetic_seq_items)/sizeof(int));
    Sequence s2(geometric_seq_items, sizeof(geometric_seq_items)/sizeof(geometric_seq_items[0]));

    cout << "### Sequence 1" << endl;
    s1.Print();

    cout << "### Sequence 2" << endl;
    s2.Print();

    Sequence s3(s1);
    s3.Concatenate(s2);

    cout << "### Sequence 3" << endl;
    s3.Print();

    return 0;
}
```

실행 예

```
### Sequence 1
수열 종류: 등차수열
수열 항: 1 3 5 7 9
### Sequence 2
수열 종류: 등비수열
수열 항: 1 2 4 8 16
### Sequence 3
수열 종류: 기타수열
수열 항: 1 3 5 7 9 1 2 4 8 16
계속하려면 아무 키나 누르십시오 . . .
```

기프트 카드를 나타내는 클래스 GiftCard를 작성하시오.

1. private 멤버 변수

- int m_balance; 기프트 카드의 잔액을 나타냄
- int m_nApproval; 기프트 카드의 결재 횟수를 나타냄.
- int m_nCharge; 기프트 카드의 충전 횟수를 나타냄.
- int *m_pChargeArr; 기프트 카드의 충전 금액을 동적으로 할당 해서 저장하기 위한 포인터
- int *m_pApprovalArr; 기프트 결재 금액을 동적으로 할당해서 저장하기 위한 포인터.

2. 생성자/소멸자

- GiftCard(void) : 잔액을 100,000 원으로 만들어주고 충전 횟수와 결재 횟수를 0으로 초기화 해준다. m_pChargeArr와 m_pApprovalArr는 각각 크기가 100인 동적 배열을 생성 해준다. 이때 초기화 리스트(Initialization List)를 이용하여 m_balance = 100,000, m_nCharge = 0, m_nApproval = 0 초기화
- GiftCard(int balance) : balance 변수 만큼의 잔액을 만들어주고 나머지는 GiftCard(void)와 같은 동작을 한다. 이때 초기화 리스트(Initialization List)를 이용하여 m_balance = balance 초기화.
- ~GiftCard(void) : m_pChargeArr와 m_pApprovalArr 에 할당된 메모리를 해제

3. Friend로 구현해야 할 연산자

- \+ 연산자 : 기프트 카드의 잔액과 결재 횟수, 충전 횟수, 결재금액 리스트, 충전 금액 리스트를 합친 새로운 GiftCard 객체를 생성하는 이항 연산자

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

- << 연산자 : cout으로 GiftCard의 결재 횟수, 금액, 충전 횟수, 금액을 출력하기 위한 이항 연산자

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

4. 클래스 멤버로 오버로딩해야 할 연산자

- = 연산자 : 피연산자로 입력된 GiftCard와 동일한 구성을 갖도록 대입하는 연산자

  GiftCard & operator=(const GiftCard & rhs);

- += 연산자 :기프트 카드의 잔액과 결재 횟수, 충전 횟수, 결재 금액 배열, 충전 금액 배열 값과 피연산자로 입력된 기프트 카드 값을 원소값을 더하여 저장하는 연산자.

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

- 전위 증가 ++연산자 : 기프트 카드에 잔액 10,000 원을 증가 시켜주고, 충전 횟수를 1 증가 시키고, 충전 금액 배열에 10,000원을 추가 해주는 연산자.

  const GiftCard & operator++()

5. Public 메소드

- void Charge(int ChargeMoney) 기프트 카드에 금액을 충전하는 함수, 충전 횟수를 1 증가시켜 주고, ChargeMoney만큼 잔액을 증가시켜 주고 충전 금액 배열에 충전된 금액을 추가 해준다.
- bool Approval(int ApprovalMoney) 기프트 카드 결재를 승인하는 함수. 결재 금액이 잔액 보다 많으면 아무것도 하지 않고 false를 리턴 해준다. 결재 금액이 잔액 보다 작으면 결재 횟수를 1증가 시켜 주고, 결재 금액 배열에 결재 금액을 추가하고 true를 리턴 해준다.
- int GetChargeNumber() 충전 횟수를 리턴해주는 함수 묵시적(Implicit) inline 함수로 구현
- int GetApprovalNumber() 결재 횟수를 리턴해주는 함수 명시적 (Explicit) inline 함수로 구현
- void ResetCard() 카드 잔액을 100,000만원으로 초기화 해주고 결재 횟수, 충전 횟수, 충전 내역을 삭제 해준다.

- 다음의 코드를 이용하여 생성된 GiftCard를 테스트하시오.

```
// 기프트 카드 인스턴스 생성
GiftCard c1;

c1.Charge(20000); // 20,000원 충전
c1.Charge(15000); // 15,000원 충전

c1.Approval(2000); // 2,000원 결재
c1.Approval(3000); // 3,000원 결재
c1.Approval(42000); // 42,000원 결재

cout << c1 << endl;      // c1 화면 출력

GiftCard c2;
c2 = c1;

++c2;                // 10,000원 충전

cout << c2 << endl;      // c2 화면 출력

c1.ResetCard();      // c1 리셋
c1.Approval(1500);   // 1,500원 결재

GiftCard c3(0);
c3.Charge(100);      // 100 원 충전
if(c3.Approval(20000)== false) // 20000만 결재 시도
    cout<<"결재 금액이 잔액 보다 많습니다."<<endl;

c3 = c3+ c2;             // c3와 c2를 더해준다.
cout << c3 << endl;      // c3 화면 출력
c3.ResetCard();          // c3 리셋

c3 += c1;                // c3와 c1을 더해준다.
cout << c3 << endl;      // c3 화면 출력

int nApproavalNum = c3.GetApprovalNumber(); // 결재 횟수 받아오기
int nChargeNum = c2.GetChargeNumber();      // 충전 횟수 받아오기

cout<<" c3의 결재 횟수는 : " <<nApproavalNum<<" 번 입니다."<< endl;
cout<<" c2의 충전 횟수는 : " <<nChargeNum<<" 번 입니다."<< endl;
```

출력 결과 예

```
잔액 : 88000
결재횟수 : 3
1번째 결재 금액 : 2000 원
2번째 결재 금액 : 3000 원
3번째 결재 금액 : 42000 원
충전횟수 : 2
1번째 충전 금액 : 20000 원
2번째 충전 금액 : 15000 원

잔액 : 98000
결재횟수 : 3
1번째 결재 금액 : 2000 원
2번째 결재 금액 : 3000 원
3번째 결재 금액 : 42000 원
충전횟수 : 3
1번째 충전 금액 : 20000 원
2번째 충전 금액 : 15000 원
3번째 충전 금액 : 10000 원

결재 금액이 잔액 보다 많습니다.
잔액 : 98100
결재횟수 : 3
1번째 결재 금액 : 2000 원
2번째 결재 금액 : 3000 원
3번째 결재 금액 : 42000 원
충전횟수 : 4
1번째 충전 금액 : 100 원
2번째 충전 금액 : 20000 원
3번째 충전 금액 : 15000 원
4번째 충전 금액 : 10000 원

잔액 : 198500
결재횟수 : 1
1번째 결재 금액 : 1500 원
충전횟수 : 0

 c3의 결재 횟수는 : 1 번 입니다.
 c2의 충전 횟수는 : 3 번 입니다.
```

학생을 관리하기 위한 프로그램을 구현하라.

다음 페이지에는 main.cpp 프로그램의 골격이 제시되어 있다. 실행결과가 실행 예와 같이 되도록, 다음 조건에 맞게 클래스 Student를 설계, 구현하고, main.cpp를 완성하시오.

- 학생의 이름은 길이가 최대 20이다. typedef을 사용하여 이름에 대한 type을 Name으로 정의한다.
- 학년은 FIRST, SECOND, THIRD, FOURTH의 네 종류가 있다. enum을 사용하여 각 종류가 차례대로 1, 2, 3, 4의 값을 갖도록 학년에 대한 type을 Year로 정의한다.
- 클래스 Student는 아래 메소드들을 갖는다. 다른 추가적인 메소드는 사용하면 안 됨.
  - 생성자 Student(Name s_name, Year s_year): 주어진 이름(s_name)과 학년(s_year)로 객체를 초기화한다.
  - 소멸자 Student(): 객체 소멸 시 필요한 작업을 수행한다.
  - Print(): 객체의 내용을 출력한다.
  - GetNumStudents(): 현재 만들어져 있는 Student 객체들의 수를 리턴한다.

main.cpp 파일

```
int main()
{
    // 1. 객체를 사용하지 않고 GetNumStudents() 메소드를 호출하여 Student 객체의 수 출력

    // 2. 세 명의 학생 (철수, 1학년), (영희, 2학년), (길동, 3학년)을 저장하는 Student 배열을 선언하고 초기화

    // 3. 배열에 있는 학생들을 출력


    // 4. 학생 (둘리, 4학년)을 저장하기 위하여 동적으로 Student 객체 생성하고 포인터 ptr이 이를 가리키게 함

    // 5. ptr 포인터를 통해 GetNumStudents()를 호출하여 Student 객체의 수 출력

    // 6. 둘리에 대한 객체 제거

    // 7. Student 객체의 수 출력

    return 0;
}
```

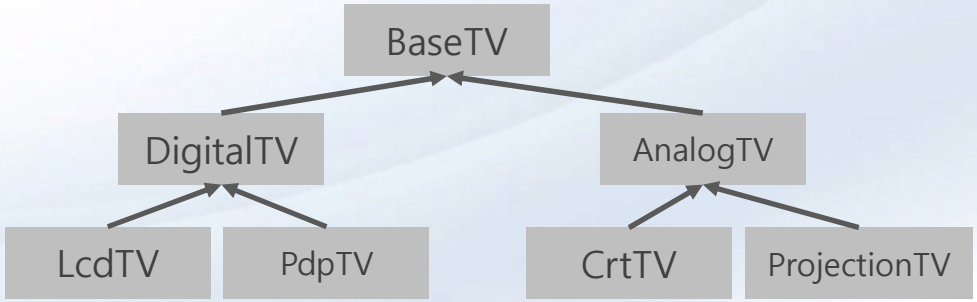
실행 예

```
현재 학생 수: 0
배열에 있는 학생 명단
이름: 철수, 학년: 1
이름: 영희, 학년: 2
이름: 길동, 학년: 3
현재 학생 수: 4
현재 학생 수: 3
계속하려면 아무 키나 누르십시오 . . .
```

시중에 판매되는 TV(텔레비전) 관리를 위한 클래스들을 구현하라.

1. 아래와 같은 클래스 상속 구조를 가지는 클래스들을 구현하라.
   1. BaseTV 는 최상위 클래스로 모든 텔레비전이 공통적으로 가지는 정보를 멤버 변수로, 공통적인 함수를 가상 (virtual) 메소드로 가짐.
   2. DigitalTV는 디지털 TV의 상위 클래스로 디지털 TV의 해상도를 , AnalogTV는 아날로그 TV의 상위 클래스로 아날로그 수신 방식을 멤버변수로 가짐.
   3. 최하위 TV클래스들은 각각의 TV가 가지고 있는 특징을 저장하는 변수를 가짐.
2. 각 클래스의 정확한 변수, 메소드 리스트는 다음장의 표 참조
3. 각각의 클래스들의 생성자는 기본 생성자만 가지고 있고 기본 생성자를 를 적절히 만든다.
4. 메인함수는 다다음장의 코드를 따름. 메인함수에서는 최상위 클래스인 BaseTV 클래스의 포인터 변수만 선언하고 최하위 클래스들을 동적 바인딩(dynamic binding) 하여 사용할 수 있게 클래스들을 구성해야 함.

상속구조



클래스 정보

BaseTV	char *mBrandName; 상표(최대 크기 100자로 동적 할당) int mPrice; 가격 원 단위 int mScreenSize; 화면 크기 inch단위 void printTVInfo(); 위 3가지 TV정보를 출력하는 함수, 가상함수 void setBrandName(char *BrandName); 상표 이름을 설정하는 함수, strcpy를 이용해서 문자열을 복사한다. void setPowerConsumption(int PowerCS); 전력 소비량을 설정하는 함수 void setPrice(int Price); 가격을 설정하는 함수 void setScreenSize(int ScreenSize); 화면 크기를 설정하는 함수
DigitalTV	bool mHangWall; 벽걸이 기능이 지원되는지 아닌지 저장하는 변수, true 지원, false 미 지원 Bool mSmartTV; 스마트 TV 기능이 지원되는지 아닌지 저장하는 변수, true 지원, false 미 지원 void setHangWall(bool hangwall); 벽결이 기능이 지원되는지 아닌지 설정하는 함수 Void setSmartTV(bool smarttv); 스마트 TV 기능이 지원되는지 아닌지 설정하는 함수 void printTVInfo(); 벽걸이 기능이 지원되는지 스마트 TV인지 출력하는 함수, overriding 함수
AnalogTV	bool mDigitalSupport; 디지털 방송 수신을 지원하는지 아닌지 저장하는 변수, true 지원 false 미지원 void seDigitalSupport(bool digitalsupport); 디지털 방송 수신 기능이 지원되는지 아닌지 설정하는 함수 void printTVInfo(); 디지털 방송 수신을 지원하는 아닌지 출력하는 함수, overriding 함수
LcdTV	bool mIsLED; LED 백라이트인지 아닌지를 저장하는 변수, true LED, false LED아님 bool mIs3DTV; 3D TV기능을 지원하는지 아닌지를 저장하는 변수, true 3D 지원, false 미지원 void setLED(bool led); LED 기능이 지원되는지 아닌지 설정하는 함수 void set3DTV(bool tdtv); 3D TV 기능이 지원되는지 아닌지 설정하는 함수 void printTVInfo(); 상위 동명 함수를 사용하여 LCD TV의 모든 정보 출력
PdpTV	bool mAcPDP; AC PDP인지 DC PDP인지 저장하는 변수, true AC PDP, false DC PDP Void setAcPDP(bool acpdp); AC PDP인지 DC PDP인지 설정하는 변수 void printTVInfo(); 상위 동명 함수를 사용하여 PDP TV의 모든 정보 출력
CrtTV	bool mIsFlat; 완전 평면인지 아닌지 저장하는 변수, true 완전 평면, false 볼록 void setFlat(bool flat); 완전 평면인지 아닌지 설정하는 함수 void printTVInfo(); 상위 동명 함수를 사용하여 CRT TV의 모든 정보 출력
ProjectionTV	int mProjectionMethod; 프로젝션 방식을 저장 0값은 DLP형, 1값은 LCD형 , 2 값은 CRT형 프로젝션 방식을 나타낸다. void setProjectionMethod(int pMethod); 프로젝션 방식을 설정하는 함수 void printTVInfo(); 상위 동명 함수를 사용하여 Projection TV의 모든 정보 출력

Main.cpp 파일

```
int main( )
{
    BaseTV *myTVs[4];

    LcdTV lcdTV;
    lcdTV.setBrandName("LG");
    lcdTV.setPrice(1000000);
    lcdTV.setScreenSize(60);
    lcdTV.setHangWall(true);
    lcdTV.setSmartTV(false);
    lcdTV.setLED(false);
    lcdTV.set3DTV(false);
    myTVs[0] = &lcdTV;

    PdpTV pdpTV;
    pdpTV.setBrandName("Samsung");
    pdpTV.setPrice(500000);
    pdpTV.setScreenSize(42);
    pdpTV.setHangWall(false);
    pdpTV.setSmartTV(true);
    pdpTV.setAcPDP(false);
    myTVs[1] = &pdpTV;

    CrtTV crtTV;
    crtTV.setBrandName("Sony");
    crtTV.setPrice(100000);
    crtTV.setScreenSize(21);
    crtTV.setDigitalSuppoirt(true);
    crtTV.setFlat(true);
    myTVs[2] = &crtTV;

    ProjectionTV prjTV;
    prjTV.setBrandName("Sharp");
    prjTV.setPrice(200000);
    prjTV.setScreenSize(54);
    prjTV.setDigitalSuppoirt(false);
    prjTV.setProjectionMethod(0);
    myTVs[3] = &prjTV;

    int i;
    for(i=0; i< 4; i++){
        myTVs[i]->printTVInfo( );
        cout<<endl;
    }
    return 0;
}
```

출력 결과

```
LCD TV
상표 : LG
가격 : 1000000 원
화면 크기 : 60 Inch
벽걸이 가능
스마트TV 기능 미지원
LED 백라이트 미지원
3D TV 기능 미지원

PDP TV
상표 : Samsung
가격 : 500000 원
화면 크기 : 42 Inch
벽걸이 불가능
스마트TV 기능 지원
DC PDP TV

CRT TV
상표 : Sony
가격 : 100000 원
화면 크기 : 21 Inch
디지털 방송 수신 가능
완전평면 아날로그 TV

Projection TV
상표 : Sharp
가격 : 200000 원
화면 크기 : 54 Inch
디지털 방송 수신 불가
DLP 프로젝션 방식
```

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]_ex02_학번
    - Ex) [A]_ex02_2012345678