Object Oriented Programming

Programming Assignment 1

Due: 2017/03/19

Announcement

1. 제출 기한은 3월 19일 자정까지입니다. LMS 시스템에서는 11시 58분 쯤부터는 업로드가 안 되니, 유의해주시기 바랍니다.
2. 과제는 늦게 제출하면 이유를 불문하고 0점입니다.
3. 채점은 Visual Studio 2015 환경에서 이루어집니다. 파일을 업로드하실 때, 작업하신 환경에 있는 프로젝트 폴더를 그대로 압축해서 올려주십시오. 그리고 폴더명은 문제1_학번의 형태로 만들어주십시오.

예시) `Problem1_20162939, Problem2_20162939, Problem3_20162939`

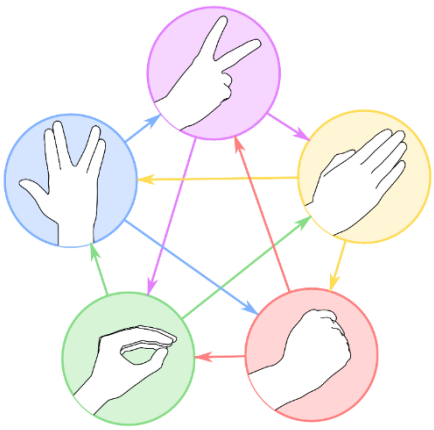
4. 모든 문제는 file 입출력이 필수입니다. input 파일은 input.txt 그리고 output 파일은 output.txt의 형태로 이루어지니, 코드를 짜실 때 그 점 유의해 주셨으면 합니다.
5. C++에서 배운 내용을 토대로 작성하셔야 합니다. C++로 구현 가능한 내용을 C로 구현했을 경우에는 감점으로 처리합니다.

예시) `scanf, printf, fprintf, fscanf, malloc, free`

1번 문제

9명의 학생들이 가위바위보의 확장 게임인 '가위-바위-보자기-스폭-도마뱀'을 한다고 가정하자. 가위바위보가 가위, 바위, 보자기를 가지고 진행이 된다면, 이 게임은 가위, 바위, 보자기, 스폭, 도마뱀 다섯 개로 진행이 되고, 규칙은 다음과 같다.

>가위는 보자기를 이긴다.
>가위는 도마뱀을 이긴다.
>보자기는 바위를 이긴다.
>보자기는 스폭을 이긴다.
>바위는 가위를 이긴다.
>바위는 도마뱀을 이긴다.
>도마뱀은 스폭을 이긴다.
>도마뱀은 보자기를 이긴다.
>스폭은 가위를 이긴다.
>스폭은 바위를 이긴다.



각각의 학생들은 다음과 같은 전략을 가지고 '가위-바위-보자기-스폭-도마뱀' 게임에 임한다.
학생1: 가위만 낸다.
학생2: 바위만 낸다.
학생3: 보자기만 낸다.
학생4: 도마뱀만 낸다.
학생5: 스폭만 낸다.
학생6: 가위와 바위 중 임의로 선택해서 낸다.
학생7: 보자기, 도마뱀과 스폭 중 임의로 선택해서 낸다.
학생8: 가위, 바위, 보자기 중 임의로 선택해서 낸다.
학생9: 가위, 바위, 보자기, 도마뱀, 스폭 중 임의로 선택해서 낸다.

이 때, 여러 학생이 '가위-바위-보자기-스폭-도마뱀'을 했을 때, 승자를 찾는 프로그램을 구현해보자. `input`은 대결하는 두 학생들의 목록으로 주어진다. 예를 들어,

3
1 2 7

로 주어졌다고 하자. 그러면 첫 줄은 몇 명의 학생이 대결에 참여하는가를 나타낸다. 그 다음 줄은 해당 학생들의 `index`가 주어진다. 이 경우에는 학생1, 학생2과 학생7이 대결하고, `output`은

2

와 같이 출력한다. 만약 '가위-바위-보자기-스폭-도마뱀'을 했을 때, 승부가 나지 않으면 0으로 표시한다.

다음은 구현할 때에 필요한 사항들이다.
1. 가위, 바위, 보자기, 도마뱀, 그리고 스폭은 `enum SRPSL`의 형태로 구현한다.

2. 학생들이 가위, 바위, 보자기, 스폭, 도마뱀 중 무엇을 낼 것인지를 결정하는 과정은 `enum SRPSL ChooseAmongSRPSL(int index)` 함수로 표현한다. 이 함수는 학생의 `index`를 인자로 받아서 그 학생이 가위, 바위, 보자기, 스폭과 도마뱀 중에서 무엇을 낼 지 결정하여 `return` 한다.
3. '가위-바위-보자기-스폭-도마뱀'의 대결 결과는 `bool IsWinner(enum SRPSL A, enum SRPSL B)` 함수의 형태로 구현한다. 이 함수는 A와 B가 '가위-바위-보자기-스폭-도마뱀' 게임으로 대결했을 때, 어느 인자가 이기는 지를 계산한다. 그리고 A가 이기면 `true`을 반환하고, B가 이기면 `false`를 반환하도록 한다.
4. `input.txt`에 기록되어 있는 내용을 입력받을 때, 동적 배열을 생성해서 입력 받는다. 그리고 '가위-바위-보자기-스폭-도마뱀 게임에 참가하는 학생의 수에 맞게 참가하는 학생들의 index와 각 학생들이 가위, 바위, 보자기, 스폭 그리고 도마뱀 중에서 무엇을 냈는지를 동적 할당을 통해서 저장하도록 한다. 다음 학생들의 대결을 진행하기 전에 동적 배열을 해제한다. input file이 EOF(end of file)에 도달할 때까지 배열의 할당과 해제 과정을 반복한다. 동적 배열의 할당과 해제는 각각 `new` 연산자와 `delete` 연산자로 구현해야 한다.

예제

Input	Output
5 6 8 9 1 2 3 1 2 3 2 1 6	2 6 8 9 0 6

채점 기준

1. Accuracy - 40 pts
2. Program Structure
    1. (5 pts) `File I/O` 구현 (`Format`이 정확하지 않을 경우 감점)
    2. (5 pts) `enum SPRSL` 구현
    3. (10 pts) `enum SRPSL ChooseAmongSRPSL(int index)` 구현
    4. (10 pts) 동적 배열 구현 (`new`와 `delete`를 이용하지 않을 경우 감점)
    5. (10 pts) `bool IsWinner(enum SRPSL A, enum SRPSL B)` 함수 구현
3. Code Readability - 10 pts
4. Comment - 10 pts

2번 문제

학생 A는 수를 거꾸로 읽는 버릇이 있다. 학생 B는 학생 A에게 두 수를 주고, 더 큰 수를 말해보라고 했다. 예를 들어, 552와 193이 주어지면, 각각 255와 391로 읽기 때문에 391이 더 크다고 이야기를 한다.

두 수가 주어졌을 때, 학생 A의 대답을 출력하는 프로그램을 C++ string function을 이용하여 작성하시오. String 기본 method들은 <string>을 include하면 사용할 수 있다. 라이브러리를 참조하여 기본 method들을 활용해 구현해야 한다.

- **string reverseNum(string num) ;**

  한 개의 string을 뒤집은 string을 리턴하는 함수.
  Ex) reverseStr("1230") = "0321"

- **string removeZero(string num) ;**

  수를 뒤집은 string이 입력되었을 때 맨 앞자리부터 연속된 0을 모두 제거하여 string을 리턴하는 함수. Ex) removeZero("005031") = "5031"

- **string compareNum(string num1, string num2) ;**

  뒤집은 두 개의 string이 입력되었을 때 더 큰 수를 리턴하는 함수.
  Ex) compareNum("437", "398") = "437"

입력

Input.txt로 받아들인다. Input.txt 파일의 첫째 줄에 학생 B가 칠판에 적은 두 수 A와 B가 주어진다. 두 수는 같지 않다.

출력

Output.txt로 출력한다. 각 줄마다 학생 A의 대답을 출력한다.

예제

Input	Output
734 893 139 1350	437 931

채점 기준

1. Accuracy - 35 pts
2. Program Structure

    1. (5 pts) `File I/O` 구현 (`Format`이 정확하지 않을 경우 감점)
    2. (10 pts) `reverseStr()` 구현 (`string`을 이용하지 않을 경우 감점)
    3. (10 pts) `removeZero()` 구현 (`string`을 이용하지 않을 경우 감점)
    4. (10 pts) `compareNum()` 구현 (`string`을 이용하지 않을 경우 감점)
    5. (5 pts) `String method` 활용 추가 점수
3. Code Readability - 10 pts
4. Comment - 15 pts

참고
String method를 확인하고 싶다면 다음 사이트를 활용할 수 있다.
http://www.cplusplus.com/reference/string/string/

3번 문제

아래의 구조체와 함수를 구현하여 10진수의 유리수를 n진수로 변환하는 프로그램을 작성하여라. (n = 2,3,…,9)

```
struct DecimalConversion {
   int Integer; //10진수의 정수 부분을 저장한다.
   float Fraction; //10진수의 소수 부분을 저장한다.
   string ConvertedNumber; //변환될 n진수를 string으로 저장한다.
};
```

- **void decompose (float num, struct DecimalConversion& dc);**

  변환할 10진수 num을 정수 부분과 소수 부분으로 나누어 구조체 dc의 Integer와 Fraction에 저장한다.

  <Hint> C++ Casting을 이용하면 정수 부분을 구할 수 있다.

- **void convert_integer (int n, struct DecimalConversion& dc);**

  dc의 Integer를 n진수로 변환하여 ConvertedNumber에 저장한다.

  <Hint> string library의 to_string()과 + operator를 이용하여라.

- **void convert_fraction (int n, struct DecimalConversion& dc);**

  dc의 Fraction를 n진수로 변환하여 ConvertedNumber에 저장한다.

  <Hint> string library의 to_string()과 + operator를 이용하여라.

- **void print_converted_number (struct DecimalConversion& dc);**

  dc의 ConvertedNumber를 output file에 저장한다.

- **void convert (float num, int n, struct DecimalConversion& dc);**

  decompose(), convert_integer(), convert_fraction()로 10진수 num을 n진수로 변환한다.

main함수는 input.txt파일을 읽고, 각각의 수를 n진수로 변환한 후 output.txt에 저장한다.

예외적으로, 10진수의 유한소수가 n진수로 변환했을 때, 무한소수가 되는 경우가 존재한다. 따라

서, 변환했을 때, 소수점 10번째 자리까지 계산해도 계산이 끝나지 않으면 예외처리를 통하여 변환된 숫자 대신 infinite number를 출력하도록 하여라.

예제

input.txt	output.txt
2 10.25 5 21.2 9 83 7 0.5	1010.01 41.1 102.0 infinite number

채점기준

1. Accuracy - 30 pts
2. Program Structure
    1. (5 pts) File I/O 구현 (Format이 정확하지 않을 경우 감점)
    1. (5 pts) convert() 구현
    2. (10 pts) decompose() 구현 (C++ casting을 이용하지 않으면 감점)
    3. (10 pts) convert_integer() 구현
    4. (15 pts) convert_fraction() 구현 (try-catch를 이용하지 않으면 감점)
    5. (5 pts) print_converted_number() 구현
3. Code Readability - 10 pts
4. Comment - 10 pts