

# **2004 TCNJ High School Programming Contest**

## **CONTEST PROBLEMS**

***DO NOT  
OPEN THIS PACKET  
UNTIL THE CONTEST STARTS  
AND YOU ARE INSTRUCTED TO DO SO!***

## Problem 1 – Factorial Madness

### ***Problem Statement:***

---

Compute the precise value of  $n!$  ( $n$  factorial) defined as the product of the positive integers from 1 to  $n$ . Continue processing input values representing  $n$  until a negative sentinel value is provided as input.

### ***Input:***

---

Input will be a non-empty list of integer values, one per line from the input file (“p1.txt”). All input values will be between 1 and 30, inclusive, with the exception of the last value in the list, which will be a negative integer indicating the end of the input.

### ***Output:***

---

For each value of  $n$ , print the precise value of  $n!$  to the standard output stream. Each output value should be printed on its own line (as shown in the example below).

### ***Example:***

---

<b>Input:</b>	<b>Output:</b>
10	3628800
35	1033314796638614492966665133753200000000
1	1
-1	

## Problem 2 – Poker

### ***Problem Statement:***

---

Five people are playing a game called “poker,” where each player is dealt a hand of five cards. They are using a standard deck of 52 card, comprised of 13 ranks (2-10, Jack, Queen, King, Ace) in 4 suits (clubs, diamonds, hearts, and spades).

Below are all possible hands, ranked from best outcome to worst. Determining the relative value of two or more hands in the same category is governed by specific rules, also given below. It is possible for two or more hands to “tie,” depending on the situation.

*Straight Flush* – Five cards of the same suit, in sequence

A hand in this category is valued by the rank of the high card. If those ranks are equal, the hands tie.

*Four of a Kind* – Four cards of the same rank

A hand in this category is valued by the rank of the four cards.

*Full House* – Three cards of one rank and two cards of another rank

A hand in this category is valued by the rank of the three cards of one rank.

*Flush* – Five cards of the same suit, not in sequence

A hand in this category is valued by the rank of the high card. If two or more players both have the same highest-ranking card, the next highest card is examined, and so on. If all cards have the same rank, the hands tie.

*Straight* – Five cards in sequence, not all of the same suit

A hand in this category is valued by the rank of the high card. If those ranks are equal, the hands tie.

*Three of a Kind* – Three cards of the same rank

A hand in this category is valued by the rank of the three cards of one rank.

*Two Pair* – Two distinct pairs of different ranks

A hand in this category is valued by the rank of the highest pair. If those ranks are equal, the second pair is examined. If those ranks are equal, the fifth card is examined. If those ranks are equal, the hands tie.

*One Pair* – One pair of the same rank

A hand in this category is valued by the rank of the pair. If those ranks are equal, the high card of the remaining three is examined. If those ranks are equal, the next highest card is examined, and so on. If the ranks of all remaining cards are equal, the hands tie.

*No Pair* – No pairs of the same rank

Any two hands in this category are valued by the rank of the highest card. If those ranks are equal, the next highest card is examined, and so on. If all cards are equal, the hands tie.

An ace may be used on either end of a straight or straight flush (after the king or before a two). If an ace is used “before a two,” the high card of the straight or straight flush is the 5.

Write a program that repeatedly examines five hands and determines the winner. Assume the cards are collected and shuffled after each play.

***Input:***

---

The first line of the input file (“p2.txt”) will contain a single positive integer representing the number of plays. For each play, read the five hands and determine the winner. Consider the first hand in each play to belong to Player 1, the second to Player 2, etc.

Each play will be represented by five input lines, where each line represents one five-card hand. Each card is represented by a two-character sequence. The first character represents the suit (C for clubs, D for diamonds, H for hearts, and S for spades). The second character represents the rank (2-9, or T for 10, J for jack, Q for queen, K for king, and A for ace). Therefore, the 4 of clubs would be represented by C4. Each card in a hand is separated by one space, with no leading spaces.

***Output:***

---

Print a header message for each play and separate plays with a single blank line. Produce output consistent with the example below. Note carefully the capitalization, spelling, and formatting of the output produced.

For each play, print the category (see above) each player’s hand was valued at and append the word “WINNER!” to the winning player’s message. If two or more players tie, append “WINNER!” to each player in the tie.

All output produced should be on the standard output stream.

**Example:**

---

**Input:**

```
2
S7 S4 C6 S3 D5
H5 CJ DJ C5 SJ
HT S8 DK D4 CT
C3 D2 S9 H6 HJ
D7 D8 DA S5 H7
S7 CT HQ H3 H9
CJ H7 C3 SA D7
D6 D9 D4 DQ DT
C8 S8 S5 H2 D8
S9 S4 ST SQ S6
```

**Output:**

```
PLAY 1:
Player 1 category: straight
Player 2 category: full house  WINNER!
Player 3 category: one pair
Player 4 category: no pair
Player 5 category: one pair

PLAY 2:
Player 1 category: no pair
Player 2 category: one pair
Player 3 category: flush  WINNER!
Player 4 category: three of a kind
Player 5 category: flush  WINNER!
```

## Problem 3 – Acronyms

### ***Problem Statement:***

---

The people who work for the government and for government contractors do not speak the same language normal people speak. They speak in acronym-ese. That is to say, many words or phrases are often shortened into acronyms (abbreviations). Usually, only their small fraternity of insiders can decipher their cryptic messages, which is apparently just the way they like it.

Write a program which takes some text and a dictionary of acronyms and translates the text into acronym-ese by substituting an acronym for the appropriate word or phrase whenever possible. Some acronyms stand for part of a longer phrase, which another acronym represents. In these situations, the longer acronym should be used.

### ***Input:***

---

The first line of the input file (“p3.txt”) contains two integer values. The first value represents the number of lines of text ( $n$ ) to process. The second value represents the number of acronym definitions ( $m$ ).

The next  $n$  lines contain text, up to 70 characters per line. The text will contain upper- and lower-case letters, numbers, spaces, and punctuation. No single word is split across two lines. The words comprising a phrase, however, can be split across multiple lines. There will be no more than 10 lines of text.

The next  $m$  lines are the acronym definitions, one per line. Each line begins with an acronym, composed of upper-case letters. The acronym is terminated by a colon (:), which is not part of the acronym or its definition. Immediately following the colon is the word or phrase (multiple words separated by single spaces), which defines the acronym. The definition is given using all lower-case letters. There will be no more than 10 acronym definitions. The maximum length of an acronym is 8 characters, and the maximum length of an acronym definition is 40 characters.

### ***Output:***

---

Reproduce the text (to the standard output stream) exactly, with acronyms substituted wherever possible. Print up to 70 characters per line, but don’t separate a word (or acronym) across lines. The resulting text must retain all appropriate punctuation.

**Example:**

---

**Input:**

7 7

Bob, some quick thoughts on the super sonic splicer. For your information, I have compiled some data which should be considered management only material. Unfortunately, the diffused action directive prohibits me from implementing the idea. The bottom line is that I think we should develop a second version, the super sonic splicer supreme, which surpasses all other models. Thus it is proved, I am a genius.

FYI:For your information

SSS:super sonic splicer

SSSS:super sonic splicer supreme

QED:thus it is proved

MOM:management only material

DAD:diffused action directive

BL:bottom line

**Output:**

Bob, some quick thoughts on the SSS. FYI, I have compiled some data which should be considered MOM. Unfortunately, the DAD prohibits me from implementing the idea. The BL is that I think we should develop a second version, the SSSS, which surpasses all other models. QED, I am a genius.

## Problem 4 – Intersection of Sets

### ***Problem Statement:***

---

Write a program that will read 2 series of letters from a line, place them each into a set, and print the intersection of the two sets.

### ***Input:***

---

The input file (“p4.txt”) will contain an unknown number of lines of input. Each line of input will contain two groups of letters comprised of all uppercase letters (A-Z). A single space character will separate the two groups of letters. Each group will contain at least one letter, but could contain more than the 26 letters of the alphabet (due to repeated letters). Each group will be no longer than 40 characters in length.

### ***Output:***

---

For each line of input processed, print the intersection of the two sets on a single line, surrounded by curly braces. Print the output to the standard output stream.

### ***Example:***

---

**Input:**

```
ABC DEF
AABBC EAGGHA
DEF DEF
```

**Output:**

```
{ }
{ A }
{ DEF }
```

## Problem 5 – Golf Scores

### ***Problem Statement:***

You have been hired to automate the scoring system at a local country club. Your program will process a round of golf scores kept by a foursome of golfers. At the end, it will determine the winner of the round. In golf, the winner is the player with the lowest total score for a round of 18 holes.

### ***Input:***

The input file (“p5.txt”) will be comprised of one or more data sets representing a round of golf. Data sets are listed in the input file, one after the other, with no vertical white space separating them. Individual scores are listed separated by a single space character.

The data set for each round will consist of the number of golfers (1-4) followed by their scores for each hole. Scores for each hole are listed in order of the player number. That is, if there are four golfers, the scores from the first hole are listed for player one first, then player two, then player three, and finally player four. The scores from the second hole will follow in the same sequence. Scores from all 18 holes will be listed on the same input line, each separated by a single space character.

### ***Output:***

For each round processed by the program from the input file, print the round number, and each player’s score. Denote the winning golfer(s) as shown in the example below. Print all output to the standard output stream.

### ***Example:***

Input:

```
2
4 5 5 4 2 3 4 5 2 3 3 3 3 4 2 6 5 4 3 2 3 4 4 4 3 2 4 3 5 4 4 3 4 3 3 5
3
5 5 2 3 4 4 3 5 4 3 5 5 4 5 5 4 3 5 5 4 2 4 4 5 5 3 4 6 4 5 3 4 4 3 3 3
5 5 5 4 5 4 5 4 3 1 3 4 3 4 5 5 6 2
```

Output:

Round 1:

Player 1: 63 WINNER!

Player 2: 67

Round 2:

Player 1: 71 WINNER!

Player 2: 76

Player 3: 71 WINNER!