14주차 수업

1. FILE I/O practice

1. FILE I/O practice

Description

You are given two files. “dict.txt”, and “input.txt”.
“dict.txt” is a dictionary file. All valid words are in this file. Every words are separated by newline character.
“input.txt” is a content file.

Your task is to find every non-valid words in “input.txt”, and print it into “output.txt”. Whenever you find non-valid words in content, output the word in “output.txt”. (with newline character)

Note that a word is consecutive alphabet characters, separated by non-alphabet character. Also, word comparison is case-insensitive. (Although comparison is case-insensitive, you must output non-valid words exactly same with content file)

1. FILE I/O practice

Contraints

- Word length <= 50
- Number of words in dictionary <= 100
- Number of words in content <= 5000

1. FILE I/O practice

- You can find all needed files in ‘Statement’ tab in judge web server.
- Submit **output.txt** and **source code** file.

dict_sample.txt	**input_sample.txt**	**output_sample.txt**
i	PPAP	an
ppap	I have a pen, I have an apple. Ugh! Apple	Ugh
pine	pen! I have a pen, I have pineapple. Ugh!	pineapple
apple	Pineapple pen! Apple pen, Pineapple	Ugh
pen	pen, Ugh! Pen-Pineapple-Apple-Pen!	Pineapple
have		Pineapple
a		Ugh
file		Pineapple
server		
different		
input		
on		
is		

2. Poker Game

THE FINAL

Poker Game

- Let’s make AI to play with!

Poker Game

- You need to implement a function
- **choose_command()**
- This function determines what command COMPUTER will say (RAISE/CALL/FOLD)

choose_command()

- **int choose_command(char human_cards[3][4], char computer_cards[5][4], Player* human, Player* computer)**

- human_cards = 3 cards in HUMAN’s hand
  - e.g.) human_cards[1] = ‘S10’
  - computer can see some of human’s cards!

- computer_cards = 5 cards in COMPUTER’s hand
  - e.g.) computer_cards[1] = ‘HA’

choose_command()

- **int choose_command(char human_cards[3][4], char computer_cards[5][4], Player* human, Player* computer)**

- human, computer = Player struct pointer

choose_command()

- **int choose_command(char human_cards[3][4], char computer_cards[5][4], Player* human, Player* computer)**

- return value : 0 – RAISE, 1 – CALL, 2 – FOLD

- In other words, if this function returns 0, “COMPUTER RAISE” command will be called

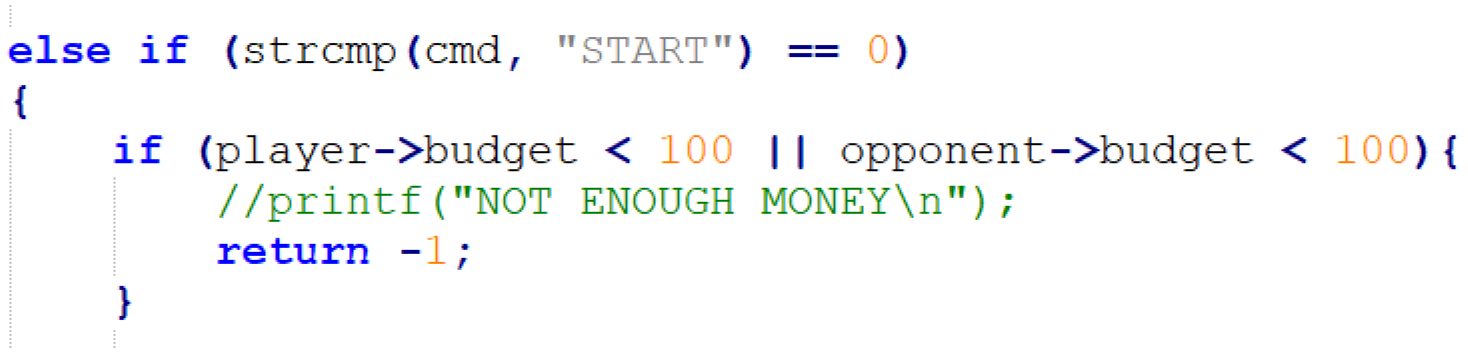
choose_command()

- **Be aware of infinite loops!**

- If you try to RAISE continuously, even if when you don’t have enough budget, loop will not end.

Minor fix on previous code

- In **call_command(),**
- Remove all printf("NOT ENOUGH MONEY");
- Instead, return -1



```
else if (strcmp(cmd, "START") == 0)
{
    if (player->budget < 100 || opponent->budget < 100){
        //printf("NOT ENOUGH MONEY\n");
        return -1;
    }
```

Skeleton Code

- **wget https://goo.gl/H4v3kd -O Task_14_2.c**

- The skeleton code is compilable, check how the game looks like!

Evaluation

- On server, your AI will compete with the AI made by TA.

- The point will be given according to your AI's win rate.
  - win rate : 0 ~ 0.3 = 0 points
  - win rate : ~ 0.5 = 20 points
  - win rate : ~ 0.6 = 40 points
  - win rate : ~ 0.7 = 70 points
  - (It will be very hard to get more than 70 points!)
  - (Don't be obsessed about getting 100 points!)
  - win rate : ~ 0.8 = 90 points
  - win rate : ~ 1.0 = 100 points

Evaluation

- What does ‘**win**’ mean?

- At the end of each game (which consists of N bets),
- If you earn more than M Won, you ‘win’.

Don’t Worry!

- TA’s AI is really stupid!
- TA’s AI only consider it’s own hand.
- If it has a good hand, it will raise a lot.
- If not, it will not bet huge money.

Announcement

- Git solution에 몇 가지 수정사항이 있습니다.

- 다음주 실습은 302동 소프트웨어 실습실(3층), 하드웨어 실습실(3층) 에서 진행됩니다.