13주차 실습

2017-11-30

13_1 Poker Betting System

- http://mrl.snu.ac.kr/courses/CourseProgrammingPractice/data/practice/2017/13_1

13_2 Horrible Array Snail : Revised

Description

Starting with the number 1 and moving to the right in a clockwise direction, a 5*5 spiral is formed as follows:

21	**22**	**23**	**24**	**25**
20	**7**	**8**	**9**	**10**
19	**6**	**1**	**2**	**11**
18	**5**	**4**	**3**	**12**
17	**16**	**15**	**14**	**13**

It can be verified that the sum of the numbers on the diagonals is 101.

Given an odd integer N, find sum of the numbers on the diagonals in a N*N spiral formed in the same way.

13_2 Horrible Array Snail : Revised

Input

Input contains an odd integer N(1 <= N < 1000).

Output

Output the sum of the numbers on the diagonals in a spiral.

Example

Input	Output
5	101
7	261

13_3 Bitcoin Master!

Description

Kwanghyun has very special ability. He can predict Bitcoin's price!

On each day, Kwanghyun can

1. Buy one Bitcoin

2. Sell any amount of bitcoin he has.

3. Do nothing

Kwanghyun has special ability, but he does not know how to get maximum profit. Given N day's information of Bitcoin price, calculate maximum profit he can achieve.

(Don't worry about Kwanghyun's budget. He is rich enough to buy Bitcoin every day.)

13_3 Bitcoin Master!

Input

On first line, number of days N is given. N <= 100,000

On second line, N integers are given. i-th integer is bitcoin price of i-th day.

(0 <= bitcoin price <= 1e9)

Output

Print one integer, maximum profit that Kwanghyun can earn.

Example

Input	Output
3 10 7 6	0
3 3 5 9	10
5 1 1 3 1 2	5

Poker Project

Step 3

What we’ve done

- Step 1 : Determining the poker hand
- Step 2 : Betting system

What we've done

- Step 1 : Determining the poker hand
- Step 2 : Betting system

→ Let's combine these!

Task 13_4 Poker Project 3/4

You need to implement 2 functions

determine_hand() and **call_command()**

Task 13_4 Poker Project 3/4

int determine_hand(char cards[5][4])

determine_hand() takes 2-dimensional char array.

e.g.) cards[0] = ‘A10’, cards[1] = ‘DQ’

return value is the score of the hand

No Pair -> return 0

One Pair -> return 1

…

Royal Straight Flush -> return 9

Task 13_4 Poker Project 3/4

int call_command(Player* human, Player* computer, char* command)

call_command takes two '**Player**' struct, and a char array

You need to use predefined '**Player'** struct
(even if you defined your own in 13-1 problem)

e.g.) command = 'HUMAN START' 'COMPUTER CALL'

return value is

- 1 if the bet ends by the command
- otherwise 0

betting end condition
-> CALL/CALL, FOLD, RAISE over opponents budget, budget < 100

Task 13_4 Poker Project 3/4

- **Skeleton Code**

- **wget** https://goo.gl/Hg83vi -O Task_13_4.c

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

Hints

- **int atoi(char *str)** [defined in stdlib.h]

- converts string to int

- e.g.)
  - char c[4] = "123"
  - int a = atoi(c) → a = 123

Hints

- **char* strtok(char *str, char* delim)** [defined in string.h]
- tokenize string by delimiter
- e.g.)
  - char* c = "Let's make Arsenal great again“;
  - char* tok = strtok(c, “ “); → tok = “Let’s”
  - char* tok2 = strtok(NULL, “ “); → tok2 = “make”
  - char* tok3 = strtok(NULL, “ “); → tok3 = “Arsenal”
- http://en.cppreference.com/w/c/string/byte/strtok