

## Practice 3 (2015/9/29)

### Friendly and Smart Interactive Query – (II)

#### 1. Another Instruction *getchar()* to Read a Character from stdin

##### A. Nightmare

```
1  #include <stdio.h>
2
3  int main() {
4      int val;
5
6      L1: printf("Please input a number that is larger than 10 and less than 100: ");
7      printf("the return value of scanf is %d\n", scanf("%d",&val));
8      if (val < 10 || val > 100) goto L1;
9      printf("Your input number is %d, this is a valid value.\n", val);
10
11     return 0;
12 }
```

☞ The above program will repeat the message display and you have no chance to input data if you input some non-numerical character → characters in keyboard buffer are not removed by *scanf()*.

##### B. Scavenger for keyboard buffer, *getchar()*

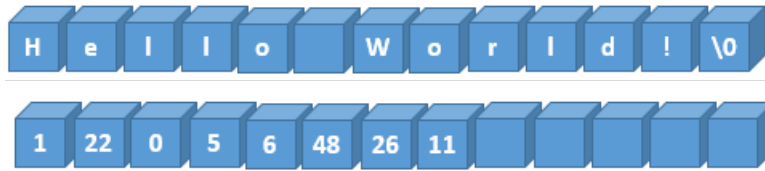
☞ You should already find *scanf()* does not receive invalid input characters and leave these invalid characters in keyboard buffer. As you try to use *scanf()* to start a new query, it easily gets into an infinite loop and you have no chance to input a character any more. Use *getchar()* to consume invalid characters of previous input. After that you can start a new query again.

**Hint:** You can use *getchar()* to remove every character in keyboard buffer until *getchar()* returns a newline character.

#### 2. String

##### A. Array Definition

☞ A group of consecutive data of the same type, for instance, integer array and character array (string).



##### B. Array Location

☞ In memory, an array is stored in consecutive locations.

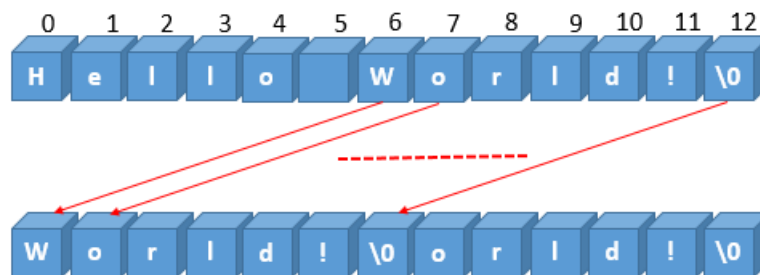
Adr	
1200	1
1204	22
1208	0
1212	5
1216	6
1220	48
1224	26
1228	11

### C. String

- ☞ A string is a sequence of characters ending with null character '\0'.
- ☞ We use null character to detect where a string ends.

### D. String Cutting

Ex. `char str[50];`



`str[0] = str[6]; str[1] = str[7]; ...; str[6] = str[12];`

### 3. Problem

Last time we already realized how to read data in different type with `scanf()`, but failed to read two integers with `scanf()` if users input non-numerical characters or string intentionally. Write a program to read two integers with a well-designed interactive query that prompts useful message to lead users to the right way of offering correct data.

- Begin a query for two integers.

```
Please input two numbers:22 101
```

- Prompt two messages for a correct input.

```
Please input two numbers:22 101
The legal number of inputs are 2
Your input is completely correct!
請按任意鍵繼續 . . .
```

- After users press a key, the program returns to initiate a new query.

```
Please input two numbers:ss 11 d33 d_
```

- d. The input program cannot recognize any integer and prompts three messages for users.

```
Please input two numbers:ss 11 d33 d
The legal number of inputs are 0
We do not receive two integers.
The garbage in keyboard buffer is "ss 11 d33 d". We just removed them from the buffer.
請按任意鍵繼續 . . .
```

- e. The program returns to initiate a new query after users press any key.

```
Please input two numbers:333 sd2 e 32_
```

- f. The program recognizes one integer and prompts three messages.

```
Please input two numbers:333 sd2 e 32
The legal number of inputs are 1
We do not receive two integers.
The garbage in keyboard buffer is "sd2 e 32". We just removed them from the buffer.
請按任意鍵繼續 . . .
```

- g. The program initiates a new query after users press any key.

```
Please input two numbers:987 348djed du3ed 343
```

- h. The program recognizes two integers but there are remaining garbage messages.

```
Please input two numbers:987 348djed du3ed 343
The legal number of inputs are 2
Although we got two integers. You also input some non-numerical characters.
The garbage in keyboard buffer is "djed du3ed 343". We just removed them from the buffer.
請按任意鍵繼續 . . .
```

**Hint:** You can use `system("cls")` to clear screen in a C program.

**Main features:**

- (a) Use `scanf()` and `getchar()` only without any flush statement to realize a friendly and smart interactive query program to accept two integers from users.
- (b) Add one features to clear the screen before starting a new iteration of query.
- (c) Use `getchar()` only rather than `scanf()` + `getchar()` to receive an integer from users. Notice: Do not put the newline character of input characters into your

garbage message, or your prompt message will look messed.

☞ At least complete features (a) and (b) before the end of practice time.

☞ The deadline to upload the program of feature (c) is the midnight of 10/5.

a. The program starts a query for one integer.

```
Please input one integer:24234324_
```

b. The program recognize one integer and there is no garbage characters.

```
Please input one integer:24234324
The input integer is 24234324
Your input is correct.
請按任意鍵繼續 . . .
```

c. The program initiates a new query after users press any key.

```
Please input one integer:erer21321 34234 2321e3d
```

d. The program cannot recognize any integer and prompt a message for this situation.

```
Please input one integer:erer21321 34234 2321e3d
Your input does not begin with a numerical character
請按任意鍵繼續 . . .
```

e. The program initiates a new query after users press any key. And it recognizes one integers but there are remaining garbage characters. Show three messages for this situation.

```
Please input one integer:83432werew 232ee ee3
The input integer is 83432
Although we got one integer from your input, your input contains non-numerical characters.
The garbage in your input is "werew 232ee ee3".
請按任意鍵繼續 . . .
```

#### 4. Bonus

The following simple program displays a weird outcome. Two floating-point numbers are displayed totally in a wrong way. Try to explain why the outcome looks like this.

```
1 #include<stdio.h>
2
3 int main(void)
4 {
5     printf("Print two floating-point numbers as two integers: %d %d \n", 8403810.434f, 10.3f);
6
7     return 0;
8 }
```

```
Print two floating-point numbers as two integers: 1073741824 -1610612736
-----
Process exited after 0.8369 seconds with return value 0
請按任意鍵繼續 . . .
```

☞ The deadline of Bonus is still the midnight. Remember to mail me your test program along with your analysis and conclusions.

☞ You have to finish features (a), (b) and (c) first and submit your program to E3. After that, you are qualified to take this challenge.

Hint: Study IEEE Std 754 first.

## 5. Further Reading

2's complement addition and subtraction, fflush(stdin), scanf("%\*[^abc]")