Day 14 Documentation

Md. Mahfuj Hasan Shohug

BDCOM0019

---

1. Exercise 4-9:

Problem: Our getch and ungetch do not handle a pushed-back EOF correctly. Decide what their properties ought to be if an EOF is pushed back, then implement your design.

Analysis:

1. In this problem I just continuing form the previous problem solution on the book exercise 4-6 and 4-7. On this problem we just do some functionality or mathematic operation but we don't handle EOF pushed back.
2. uses the standard input to read a character using the getchar() method. The function sets the newline character ('n') to the variable if the character reading value is equal to 26 (which indicates the ASCII code for Ctrl+Z, which is frequently used as an EOF indicator on some systems).

Outputs:

```
D:\Reposetory\MdMahfujHasanShohug\C&DS\Day_14\Exercise 4-9.exe
-> Enter equations in the form: "1 1 + 2 5 + *"
-> Use "a=1 press enter b=2 press enter c=3" to store variables.
-> Use "a b c * *" to use stored variables.
-----------------------------------------------
>>> Command Help:
>>>     !:      Clear memory.
>>>     p:      Print last character.
>>>     s:      Swap last two characters.
>>>     d:      Duplicate the last input.
>>>     L:      Print variable list.

5 6 + ^Z
        11
```

```
D:\Reposetory\MdMahfujHasanShohug\C&DS\Day_14\Exercise 4-9.exe
-> Enter equations in the form: "1 1 + 2 5 + *"
-> Use "a=1 press enter b=2 press enter c=3" to store variables.
-> Use "a b c * *" to use stored variables.
-----------------------------------------------
>>> Command Help:
>>>     !:      Clear memory.
>>>     p:      Print last character.
>>>     s:      Swap last two characters.
>>>     d:      Duplicate the last input.
>>>     L:      Print variable list.

^Z

--------------------------------
Process exited after 3.635 seconds with return value 0
Press any key to continue . . .
```

Source code:

Exercise 4-13.c | Exercise 4-14.c | Exercise 4-9.c

```c
}

// get a (possibly pushed back) character
// checks to see if there are any chars in buffer. If there are, get those and return it. I
int getch(void)
{
    if (bufp > 0){
        return buf[--bufp];
    }
    else{
        char new_char = getchar();
        if (new_char == 26) {
            new_char ='\n';
        }
        return new_char;
    }
}

// push character back on input
// if bufp is less than BUFSIZE, there is room to store more chars to be read by getch next
void ungetch(int c)
{
    if (bufp >= BUFSIZE)
        printf("ungetch: too many characters\n");
    else
        buf[bufp++] = c;
}

// prints the top element in the value stack
void printTop(void)
{
    if (sp > 0)
        printf("\t%.8g\n", val[sp - 1]);
    else
        printf("error: stack empty\n");
}
```

ces | Compile Log | Debug | Find Results | Close

```
- Compiler Name: TDM-GCC 4.9.2 64-bit Release

Processing C source file...
--------
- C Compiler: C:\Program Files (x86)\Dev-Cpp\MinGW64\bin\gcc.exe
- Command: gcc.exe "D:\Repository\MdMahfujHasanShohug\C&DS\Day_14\Exercise 4-9.c" -o "

Compilation results...
--------
- Errors: 0
- Warnings: 0
- Output Filename: D:\Repository\MdMahfujHasanShohug\C&DS\Day_14\Exercise 4-9.exe
- Output Size: 134.375 KiB
- Compilation Time: 0.19s
```

2. Exercise 4-13:

Problem: Write a recursive version of the function reverse(s), which reverses the String s in place.

Analysis:

1. My custom function “getInput()” have the input prompts, how takes input of any char input a string using fgets(). Which one we need to reverse.
2. The “reverse_str()” function is designed to iterate over a string. Arguments start start, index end end and string str[] are required. The function swaps the characters at the start and end indices in an iterative process to reverse the string. When the start index is equal to or equal to the end index, a base case is used to terminate the loop.
3. Here in this program input: BDCOM, “reverse_str()” function recursively give the output: MOCDB.

Outputs:

```
D:\Reposetory\MdMahfujHasanShohug\C&DS\Day_14\Exercise 4-13.exe
Enter any string: BDCOM
Your Inputted String: BDCOM
Updated reversed string: MOCDB

--------------------------------
Process exited after 3.821 seconds with return value 0
Press any key to continue . . .
```

```
D:\Reposetory\MdMahfujHasanShohug\C&DS\Day_14\Exercise 4-13.exe
Enter any string: Hello! World????
Your Inputted String: Hello! World????
Updated reversed string: ????dlroW !olleH

--------------------------------
Process exited after 10.41 seconds with return value 0
Press any key to continue . . .
```

Source Code:

Exercise 4-13.c | Exercise 4-14.c | Exercise 4-9.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAX 100
/*****************************************************************************************
** Function Name: main, getInput, reverse_str                                           **
** Inputs       : 1. argc    -- The number of parameters provided to the main function**
**              : 2. argv    -- The pointer to the input string array of parameters   **
** Variable     : str[]      -- Inputed string                                        **
**              : temp       -- store reverse char on this                            **
**              : size       --  size of string                                       **
**              : start,end  --  loop variable                                        **
** Return       : = 0        -- Success                                               **
**              : < 0        -- Failed                                                **
** Note         :recursive version of the string reversing                            **
*****************************************************************************************/
/*Function to get user input*/
void getInput(char input[], int size)
{
    fgets(input, size, stdin);
    /*Remove the trailing newline character from the input*/
    size_t input_length = strlen(input);
    if (input_length > 0 && input[input_length - 1] == '\n')
    {
        input[input_length - 1] = '\0';
    }
}

/* Function for reversing the string */
void reverse_str(char str[], int start, int end)
{
    if (start >= end)
    {
        return; /* Base case: string is already reversed or empty*/
    }

    /* Swap characters at start and end indices*/
    char temp = str[start];
    str[start] = str[end];
    str[end] = temp;

    /*Recursive call to reverse the substring between start and end*/
    reverse_str(str, start + 1, end - 1);
```

```c
}

/* Main Function*/
int main(int argc, char *argv[])
{
    char str[MAX];
    printf("Enter any string: ");
    getInput(str, sizeof(str));

    printf("Your Inputted String: %s\n", str);

    int start = 0;
    int end = strlen(str) - 1;
    reverse_str(str, start, end);

    printf("Updated reversed string: %s\n", str);

    return 0;
}
```

Compile Log | Debug | Find Results | Close

```
Compilation results...
--------
- Errors: 0
- Warnings: 0
- Output Filename: D:\Repository\MdMahfujHasanShohug\C&DS\Day_14\Exercise 4-13.exe
- Output Size: 129.1484375 KiB
- Compilation Time: 0.17s
```

3. Exercise 4-14:

Problem: Define a macro swap(t,x,y) that interchanges two arguments of type t. (Block structure will help.).

Analysis:

1. I define the macro swap in this code, which enables the values of two variables, x_val and y_val, to be switched for a specified type.
2. The macro requires three inputs: type_s, x_val, and y_val. The temporary variable of type temp is declared inside the type_s macro. Temporary variables are used to swap the values of x_val and y_val variables.
3. Use of this macro eliminates the need to write explicit swap code and offers an easy method for changing variable values. In short in this program, swap the values of X and Y. It shows how to utilize a macro to swap the values of two variables.

Outputs:

```
D:\Reposetory\MdMahfujHasanShohug\C&DS\Day_14\Exercise 4-14.exe
Enter X value: 45
Enter Y value: 65
Before Swap:->
                X = 45
                Y = 65
After Swap:->
                X = 65
                Y = 45

--------------------------------
Process exited after 4.741 seconds with return value 32
Press any key to continue . . .
```

```
D:\Reposetory\MdMahfujHasanShohug\C&DS\Day_14\Exercise 4-14.exe
Enter X value: a
Enter Y value: Before Swap:->
                X = 1
                Y = 0
After Swap:->
                X = 0
                Y = 1

--------------------------------
Process exited after 4.732 seconds with return value 30
Press any key to continue . . .
```

If user don't input the int number then Its show some other value, because in this program my data type was int.

Source Code:

Exercise 4-13.c | Exercise 4-14.c | Exercise 4-9.c

```c
/*Macro Define to swap values of x and y*/
#define swap(type_s, x_val, y_val)  \
    {                               \
        type_s temp;                \
        temp = x_val;               \
        x_val = y_val;              \
        y_val = temp;               \
    };
/*******************************************************************************
** Function Name: main                                                         **
** Inputs       : 1. argc   -- The number of parameters provided to the main function**
**              : 2. argv   -- The pointer to the input string array of parameters   **
** Variable     : x_val     -- First value                                      **
**              : y_val     -- Second value                                     **
**              : type_s    --  Type of the variable                           **
** Return       : = 0       -- Success                                          **
**              : < 0       -- Failed                                           **
** Note         :Define a macro that interchanges two arguments of type         **
*******************************************************************************/

// Main Function
int main(int argc, char *argv[])
{
    int x_val, y_val;
    printf("Enter X value: ");
    scanf("%d", &x_val); /* Input value of X*/
    printf("Enter Y value: ");
    scanf("%d", &y_val); /* Input value of Y*/

    printf("Before Swap:->\n\t\tX = %d\n\t\tY = %d\n", x_val, y_val);
    /*Swap the values using the swap macro*/
    swap(int, x_val, y_val);
    printf("After Swap:->\n\t\tX = %d\n\t\tY = %d\n", x_val, y_val);
}
```

es | Compile Log | Debug | Find Results | Close

```
- Compiler Name: TDM-GCC 4.9.2 64-bit Release

Processing C source file...
--------
- C Compiler: C:\Program Files (x86)\Dev-Cpp\MinGW64\bin\gcc.exe
- Command: gcc.exe "D:\Repository\MdMahfujHasanShohug\C&DS\Day_14\Exercise 4-14.c" -o "D:\Repositor

Compilation results...
--------
- Errors: 0
- Warnings: 0
- Output Filename: D:\Repository\MdMahfujHasanShohug\C&DS\Day_14\Exercise 4-14.exe
- Output Size: 128.6015625 KiB
- Compilation Time: 0.17s
```