Vivian Do
Nov 04 2021
CRN 88089
Lab 10 Commands Report

CSC3320 System Level Programming

Lab Assignment 10 - Post-Lab

Due at 11:59 pm on Friday, April 02, 2021

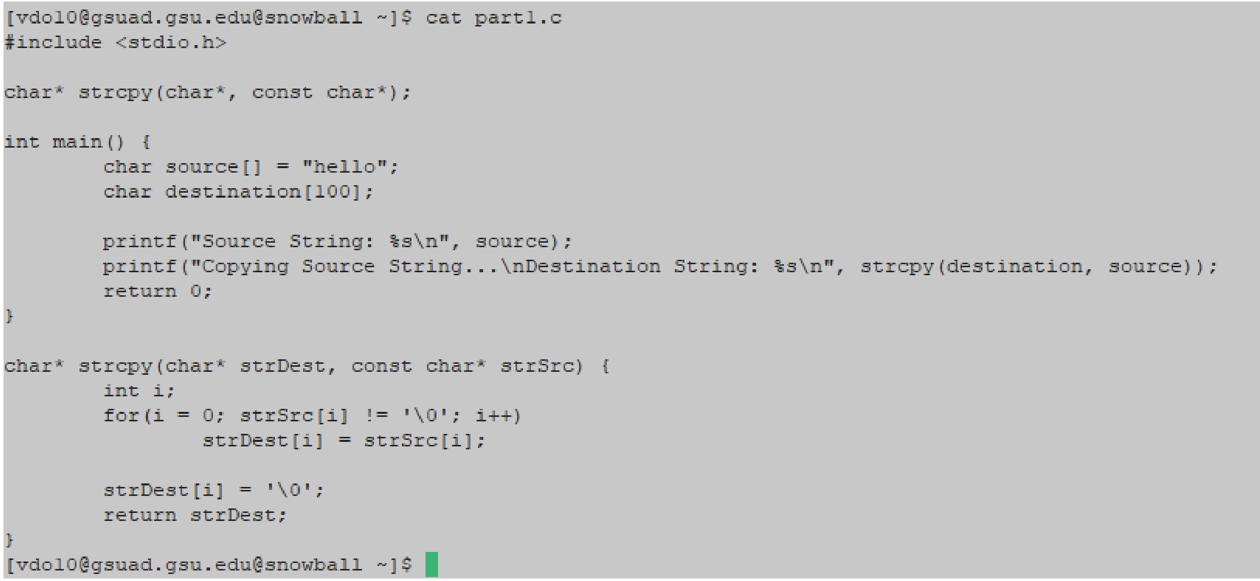
Purpose: Learn how to use the pointers to represent strings in C.

Part 1:

Write a function about string copy, the *strcpy* prototype "*char* strcpy (char* strDest, const char* strSrc)*;". Here *strDest* is destination string, *strSrc* is source string.

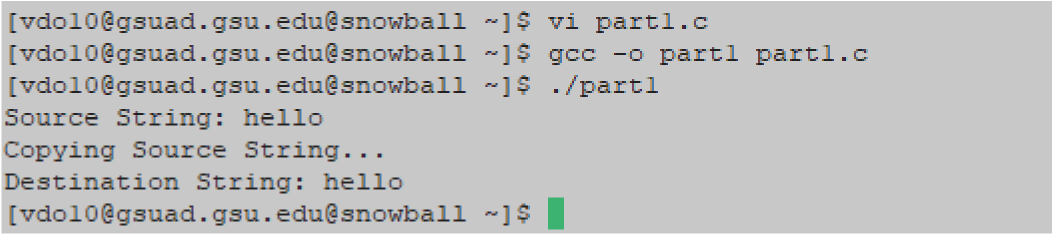
1) Write the function *strcpy*, don't call C string library.

Code:



```
[vdo10@gsuad.gsu.edu@snowball ~]$ cat part1.c
#include <stdio.h>

char* strcpy(char*, const char*);

int main() {
        char source[] = "hello";
        char destination[100];

        printf("Source String: %s\n", source);
        printf("Copying Source String...\nDestination String: %s\n", strcpy(destination, source));
        return 0;
}

char* strcpy(char* strDest, const char* strSrc) {
        int i;
        for(i = 0; strSrc[i] != '\0'; i++)
                strDest[i] = strSrc[i];

        strDest[i] = '\0';
        return strDest;
}
[vdo10@gsuad.gsu.edu@snowball ~]$
```

Output:



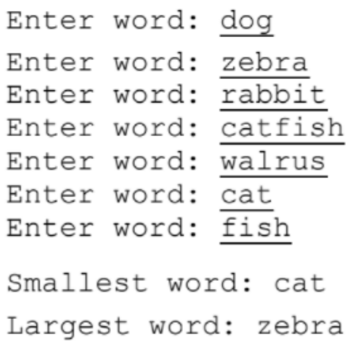
```
[vdo10@gsuad.gsu.edu@snowball ~]$ vi part1.c
[vdo10@gsuad.gsu.edu@snowball ~]$ gcc -o part1 part1.c
[vdo10@gsuad.gsu.edu@snowball ~]$ ./part1
Source String: hello
Copying Source String...
Destination String: hello
[vdo10@gsuad.gsu.edu@snowball ~]$
```

2) Here *strcpy* can copy *strSrc* to *strDest*, but why we use *char** as the return value of *strcpy*?

We use *char** as the return the reference and not the whole object when we call the array. This makes calling this function easier by using *printf* to print the destination string. This also checks if the function is working as intended.

Part 2:

Write a program ***findStr.c*** that finds the "smallest" and "largest" in a series of words. After the user enters the words, the program will determine which words would come first and last if the words were listed in dictionary order. The program must stop accepting input when the user enters a four-letter word. Assume that no word is more than 20 letters long. An interactive session with the program might look like this:



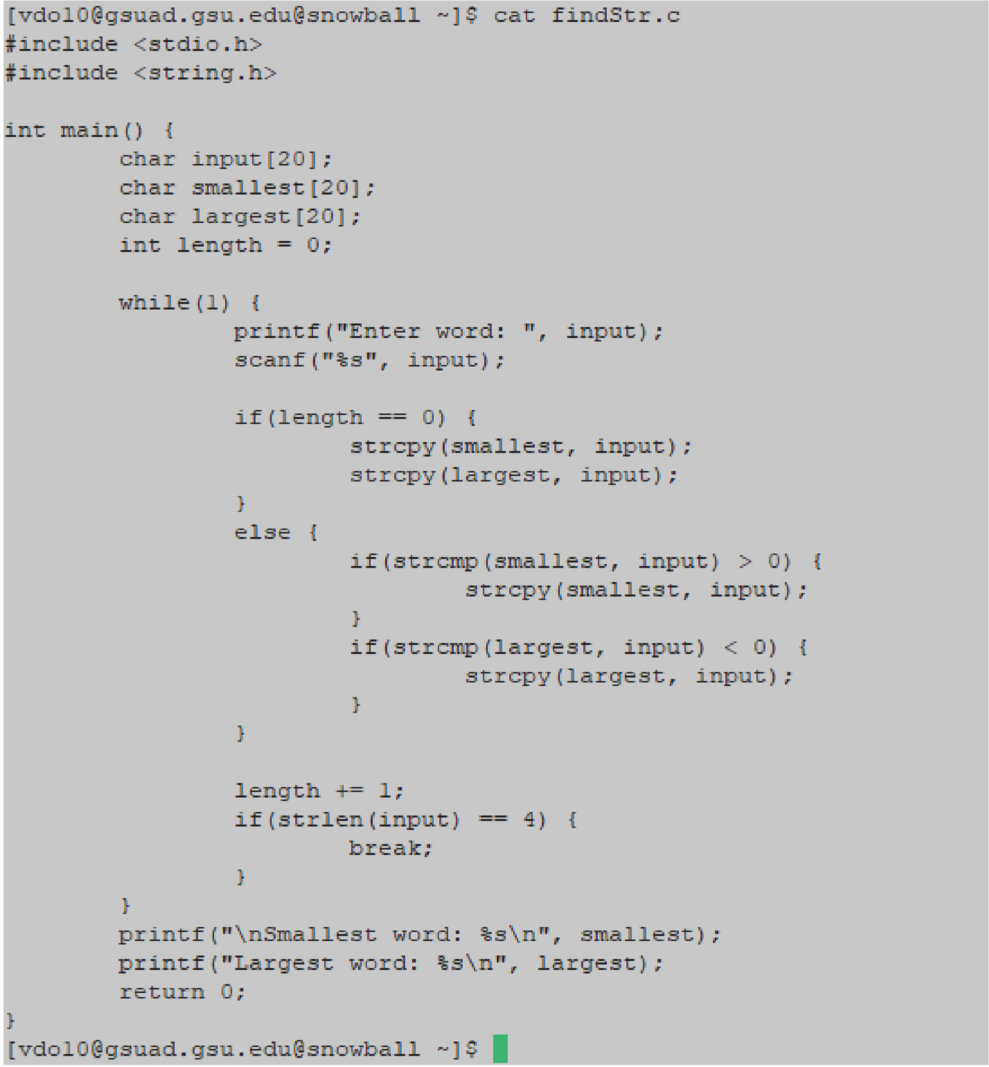
```
Enter word: dog
Enter word: zebra
Enter word: rabbit
Enter word: catfish
Enter word: walrus
Enter word: cat
Enter word: fish

Smallest word: cat
Largest word: zebra
```

Hint: Use two strings named *smallest_word* and *largest_word* to keep track of the "smallest" and "largest" words entered so far. Each time the user enters a new word, use *strcmp* to compare it with *smallest_word*; if the new word is "smaller", use *strcpy* to save it in *smallest_word*. Do a similar comparison with *largest_word*. Use *strlen* to determine when the user has entered a four-letter word.

Questions:

1) Attach the source code of your C program into the answer sheet.

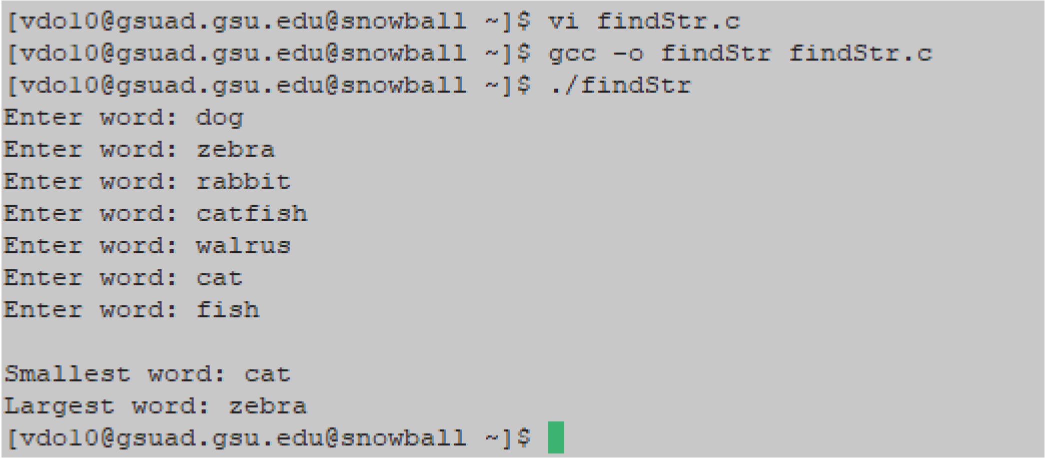
Code:



```
[vdol0@gsuad.gsu.edu@snowball ~]$ cat findStr.c
#include <stdio.h>
#include <string.h>

int main() {
        char input[20];
        char smallest[20];
        char largest[20];
        int length = 0;

        while(1) {
                printf("Enter word: ", input);
                scanf("%s", input);

                if(length == 0) {
                        strcpy(smallest, input);
                        strcpy(largest, input);
                }
                else {
                        if(strcmp(smallest, input) > 0) {
                                strcpy(smallest, input);
                        }
                        if(strcmp(largest, input) < 0) {
                                strcpy(largest, input);
                        }
                }

                length += 1;
                if(strlen(input) == 4) {
                        break;
                }
        }
        printf("\nSmallest word: %s\n", smallest);
        printf("Largest word: %s\n", largest);
        return 0;
}
[vdol0@gsuad.gsu.edu@snowball ~]$
```

2) Run the C program, attach a screenshot of the output in the answer sheet

Output:



```
[vdol0@gsuad.gsu.edu@snowball ~]$ vi findStr.c
[vdol0@gsuad.gsu.edu@snowball ~]$ gcc -o findStr findStr.c
[vdol0@gsuad.gsu.edu@snowball ~]$ ./findStr
Enter word: dog
Enter word: zebra
Enter word: rabbit
Enter word: catfish
Enter word: walrus
Enter word: cat
Enter word: fish

Smallest word: cat
Largest word: zebra
[vdol0@gsuad.gsu.edu@snowball ~]$
```

Submssion:

- Please follow the instructions below step by step, and then write a report by answering the questions and upload the report (named as Lab10_FirstNameLastName.pdf or Lab10_FirstNameLastName.doc) to Google Classroom, under the rubric Lab 10 – Post Lab Assignment.
- Upload the C files findStr.c to the folder named "Lab 10 – Post Lab" in Google Classroom.
- Please add the lab assignment NUMBER and your NAME at the top of your filesheet.