Cuong Hoang

CSC3320 System Level Programming

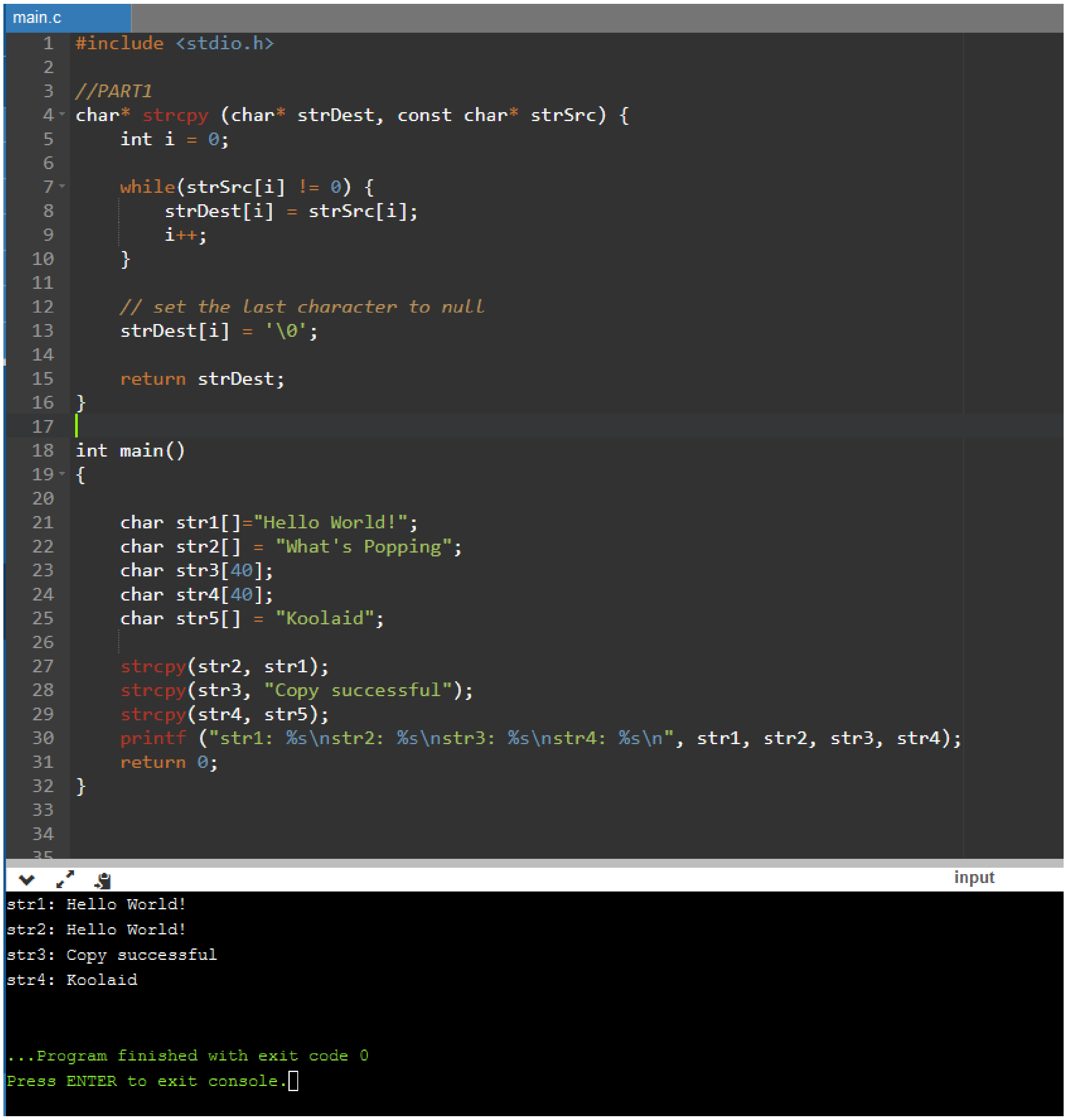
Lab Assignment 10 – Post-Lab

Part 1:

1)

```
//PART1
char* strcpy (char* strDest, const char* strSrc) {
    int i = 0;

    while(strSrc[i] != 0) {
        strDest[i] = strSrc[i];
        i++;
    }

    // set the last character to null
    strDest[i] = '\0';

    return strDest;
}

int main()
{

    char str1[]="Hello World!";
    char str2[] = "What's Popping";
    char str3[40];
    char str4[40];
    char str5[] = "Koolaid";

    strcpy(str2, str1);
    strcpy(str3, "Copy successful");
    strcpy(str4, str5);
    printf ("str1: %s\nstr2: %s\nstr3: %s\nstr4: %s\n", str1, str2, str3, str4);
    return 0;
}
```

Output:



```c
#include <stdio.h>

//PART1
char* strcpy (char* strDest, const char* strSrc) {
    int i = 0;

    while(strSrc[i] != 0) {
        strDest[i] = strSrc[i];
        i++;
    }

    // set the last character to null
    strDest[i] = '\0';

    return strDest;
}

int main()
{

    char str1[]="Hello World!";
    char str2[] = "What's Popping";
    char str3[40];
    char str4[40];
    char str5[] = "Koolaid";

    strcpy(str2, str1);
    strcpy(str3, "Copy successful");
    strcpy(str4, str5);
    printf ("str1: %s\nstr2: %s\nstr3: %s\nstr4: %s\n", str1, str2, str3, str4);
    return 0;
}
```

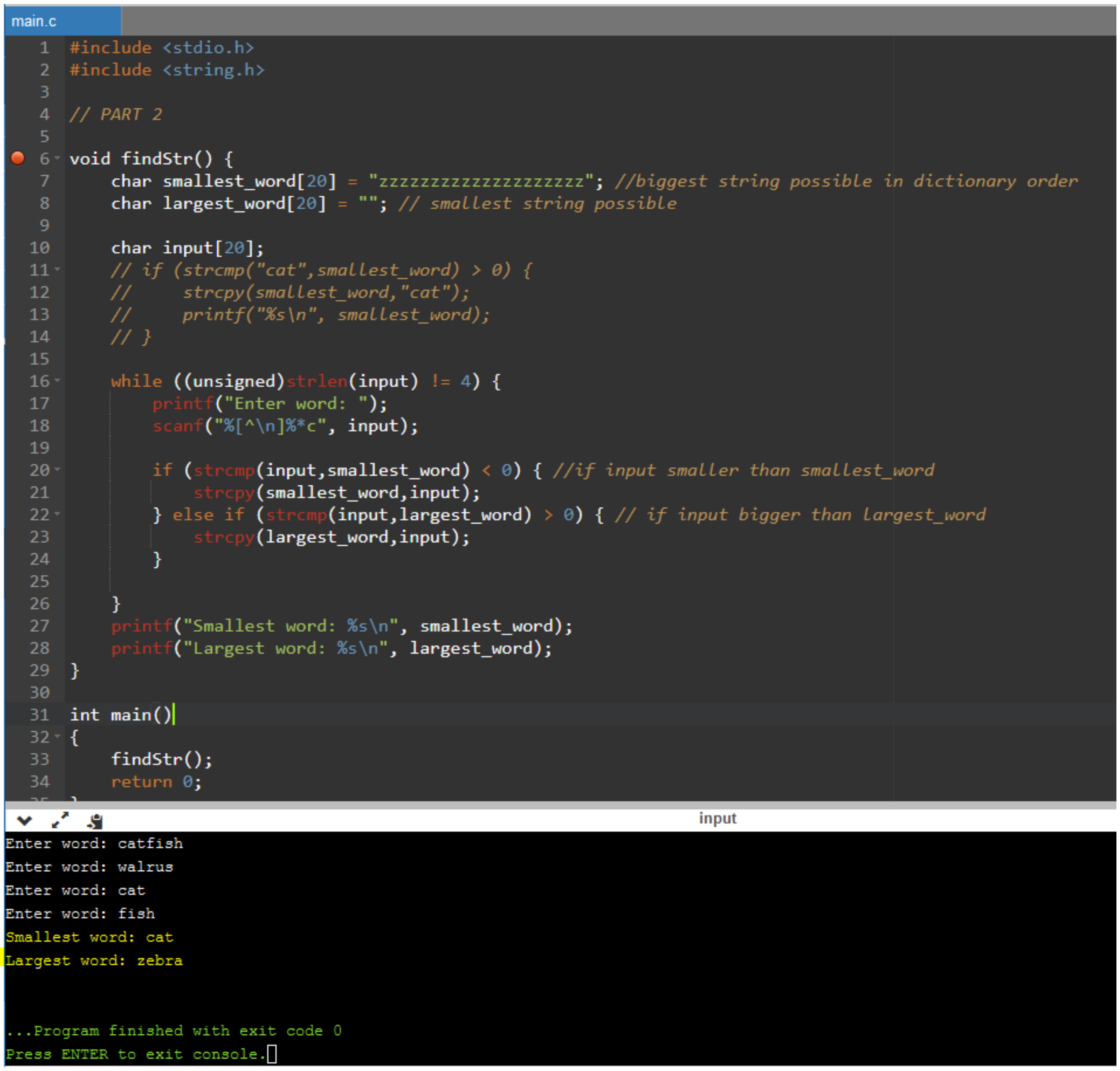
```
str1: Hello World!
str2: Hello World!
str3: Copy successful
str4: Koolaid


...Program finished with exit code 0
Press ENTER to exit console.
```

2) A pointer knows the address and value of another variable. Therefore, I think that when we passed in a pointer to the strDest that is declared from other method, we are certain that we are dealing exactly with that variable and return it as pointer to make sure we've changed the original value at the right variable (address) in our strcpy method.

Part 2:

```c
#include <stdio.h>
#include <string.h>

// PART 2

void findStr() {
    char smallest_word[20] = "zzzzzzzzzzzzzzzzzzzz"; //biggest string possible in
dictionary order
    char largest_word[20] = ""; // smallest string possible

    char input[20];
    // if (strcmp("cat",smallest_word) > 0) {
    //     strcpy(smallest_word,"cat");
    //     printf("%s\n", smallest_word);
    // }

    while ((unsigned)strlen(input) != 4) {
        printf("Enter word: ");
        scanf("%[^\n]%*c", input);

        if (strcmp(input,smallest_word) < 0) { //if input smaller than
smallest_word
            strcpy(smallest_word,input);
        } else if (strcmp(input,largest_word) > 0) { // if input bigger than
largest_word
            strcpy(largest_word,input);
        }

    }
    printf("Smallest word: %s\n", smallest_word);
    printf("Largest word: %s\n", largest_word);
}

int main()
{
    findStr();
    return 0;
}
```

Output:



main.c

```c
#include <stdio.h>
#include <string.h>

// PART 2

void findStr() {
    char smallest_word[20] = "zzzzzzzzzzzzzzzzzzz"; //biggest string possible in dictionary order
    char largest_word[20] = ""; // smallest string possible

    char input[20];
    // if (strcmp("cat",smallest_word) > 0) {
    //     strcpy(smallest_word,"cat");
    //     printf("%s\n", smallest_word);
    // }

    while ((unsigned)strlen(input) != 4) {
        printf("Enter word: ");
        scanf("%[^\n]%*c", input);

        if (strcmp(input,smallest_word) < 0) { //if input smaller than smallest_word
            strcpy(smallest_word,input);
        } else if (strcmp(input,largest_word) > 0) { // if input bigger than largest_word
            strcpy(largest_word,input);
        }

    }
    printf("Smallest word: %s\n", smallest_word);
    printf("Largest word: %s\n", largest_word);
}

int main()
{
    findStr();
    return 0;
```

input

```
Enter word: catfish
Enter word: walrus
Enter word: cat
Enter word: fish
Smallest word: cat
Largest word: zebra


...Program finished with exit code 0
Press ENTER to exit console.
```