Assignment 6

1 code

```c
#include <stdio.h>
#include <stdlib.h>
#include <strings.h>
#define MAX_LENGTH 50

char *find_blank(char *start)
{
    char *copy = start;
    while (*copy != (char)NULL)
    {
        if (*copy == ' ')
        {
            return copy;
        }
        copy++;
    }
    printf("There are no blanks\n");
    return (char *)NULL;
}
```

Main

```
// test 1

char *test1 = "test one let's see how it goes.";

printf("\n\nTest one: %s\n", test1);
printf("Address of the first character of test 1 (Should be less than the next print statement by 4): %p\n", test1);
printf("Address of the first blank of test 1: %p\n\n\n", find_blank(test1));

// test 2

char *test2 = " first char is the first blank.";

printf("Test two: %s\n", test2);
printf("Address of the first character of test 2 (Should match the next print): %p\n", test2);
printf("Address of the first blank of test 2: %p\n\n\n", find_blank(test2));

// test 3

char *test3 = "lastcharistheblank ";

printf("Test three: %s\n", test3);

char *test3copy = test3;
char *test3last = test3copy;

while (*(test3last + 1) != (char)NULL)
{
    test3last = test3copy;
    test3copy++;
    test3last++;
}

printf("Address of the last character of test 3 (Should match the next print): %p\n", test3last);
printf("Address of the first blank of test 3: %p\n\n\n", find_blank(test3));

// test 4
```

1 tests

```
Test one: test one let's see how it goes.
Address of the first character of test 1 (Should be less than the next print
 statement by 4): 00405078
Address of the first blank of test 1: 0040507C


Test two:  first char is the first blank.
Address of the first character of test 2 (Should match the next print): 0040
5138
Address of the first blank of test 2: 00405138


Test three: lastcharistheblank
Address of the last character of test 3 (Should match the next print): 00405
1F2
Address of the first blank of test 3: 004051F2


Test four: therearenoblanks
There are no blanks
```

2 code

```c
void four_stars(char *start)
{
    int i = 0;
    // char* startcopy = start;
    while (i < 4)
    {
        *(start + i) = '*';
        // startcopy++;
        i++;
    }
}
```

Main

```c
char *test5 = "Four stars";
char *test5heap = (char *)malloc(sizeof(char) * 200);
char *mover = test5heap;
while (*test5 != (char)NULL)
{
    *mover = *test5;
    mover++;
    test5++;
    if (*(test5 + 1) == NULL)
    {
        *(mover + 1) = NULL;
    }
}

printf("String before the 4 stars: %s\n\n\n", test5heap);

four_stars(test5heap);
printf("String after the 4 stars: %s\n\n\n", test5heap);

free(test5heap);
```

Test

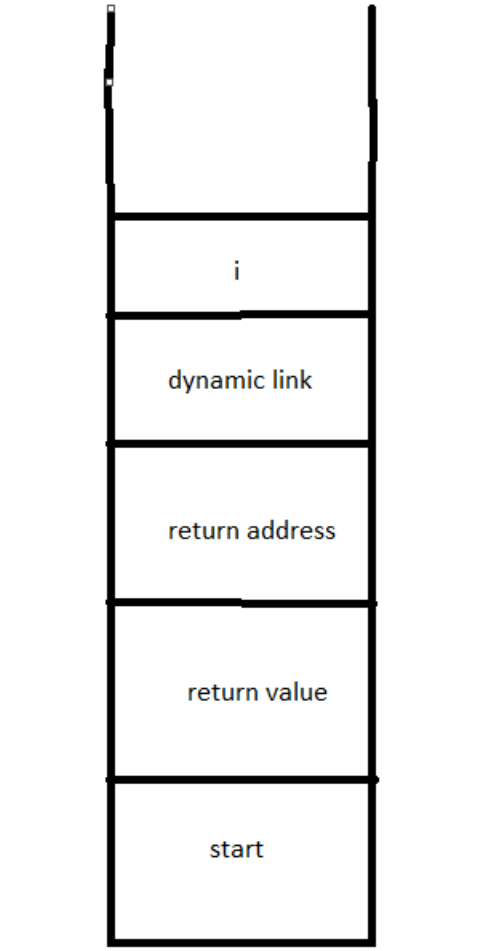
```
String before the 4 stars: Four stars

String after the 4 stars: **** stars
```

3

a

```
1         AND R0, R0, #0; i = 0
2         STR R0, R5, #0
3
4   LOOP  LDR R0, R5, #0
5         ADD R0, R0, #-4
6         BRzp NEXT
7
8
9         ; CODE TO CHANGE TO *
10
11
12        ;LOOP VARIABLE
13        LDR R0, R5, #4
14        ADD R0, R0, #1
15        STR R0, R5, #0
16        BRnzp LOOP
17
18
19  NEXT  ;WHATEVER
20
21  HALT
22  .END
```

B


i
dynamic link
return address
return value
start

4

Code

```
void censor(char *start)
{
    char *startcopy = start;
    do
    {
        start=find_blank(start);
        if ((start - startcopy == 4))
        {
            four_stars(startcopy);
        }
        start++;
        startcopy = start;
    } while (find_blank(start) != NULL);
                    (int)4
    if (*(start + 4) == (char)NULL)
    {
        four_stars(startcopy);
    }
}
```

Main

```
    char *test6 = "Final test should take notes.";
    char *test6heap = (char *)malloc(sizeof(char) * 200);
    char *mover2 = test5heap;
    while (*test6 != (char)NULL)
    {
        *mover2 = *test6;
        mover2++;
        test6++;
        if (*(test6 + 1) == NULL)
        {
            *(mover2 + 1) = NULL;
        }
    }

    printf("Test 6 before censoring: %s\n", test6heap);
    censor(test6heap);
    printf("Test 6 after censoring: %s\n", test6heap);
    printf("\n\n");
    return EXIT_SUCCESS;
```

Test

```
Test 6 before censoring: Final test should take notes.
There are no blanks
Test 6 after censoring: Final **** should **** notes.
```