

Lab Assignment 6

Akshat Mittal - 20107

July 2021

Contents

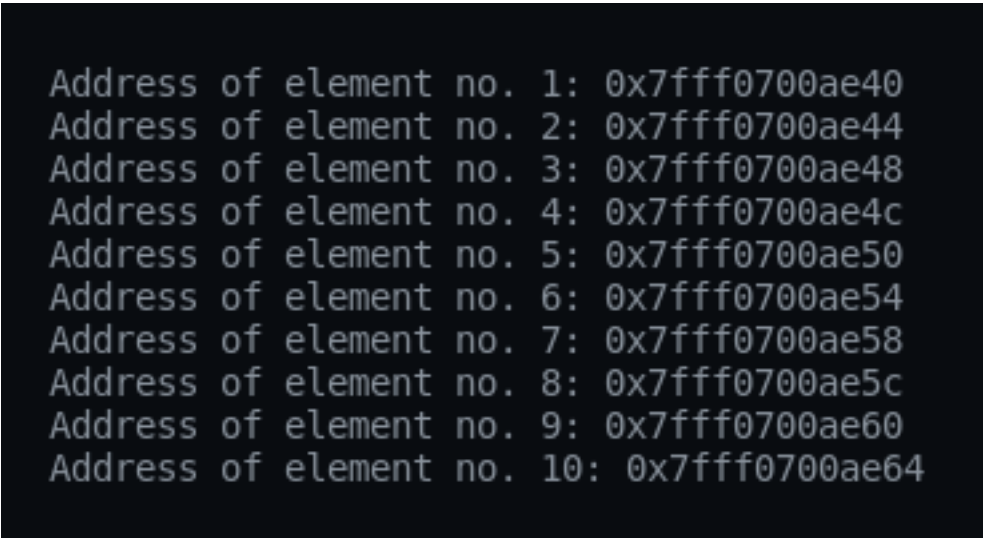
1. Addresses of array elements
2. Use of referencing and dereferencing operators
3. Circular swapping
4. Finding Factorial
5. Vowels and consonants
6. Sorting using pointers
7. Sum of elements of an array
8. Reversing the string
9. Reversing individual characters of string
10. No. of words in a string
11. Comparing two strings
12. no. of alphabets, numbers, special characters
13. Copying one string into another
14. To find maximum occurring character in a string
15. Checking for substring
16. Library Functions of string.h
17. Conversion of lowercase to uppercase
18. Fetching a substring
19. Replacing a word
20. Deleting vowels from a string

1

1.1 Code

```
#include <stdio.h>
int main()
{
    printf("\n\n\n");
    int a[10];
    for (int i = 0; i < 10; ++i)
    {
        printf("\n Address of element no. %d: %p", i + 1, &a[i]);
        // & operator gets the address of specified element
    }
    printf("\n\n\n");
    return 0;
}
```

1.2 Output



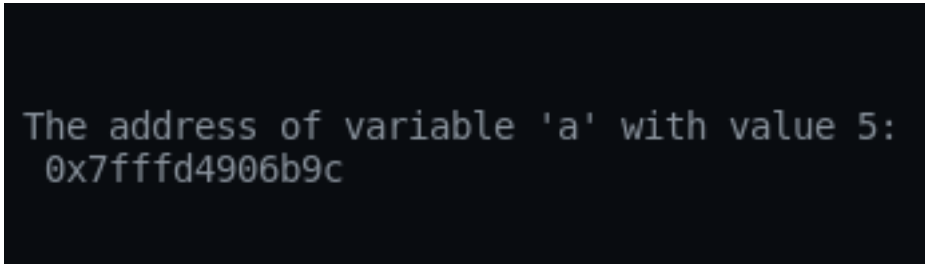
```
Address of element no. 1: 0x7fff0700ae40
Address of element no. 2: 0x7fff0700ae44
Address of element no. 3: 0x7fff0700ae48
Address of element no. 4: 0x7fff0700ae4c
Address of element no. 5: 0x7fff0700ae50
Address of element no. 6: 0x7fff0700ae54
Address of element no. 7: 0x7fff0700ae58
Address of element no. 8: 0x7fff0700ae5c
Address of element no. 9: 0x7fff0700ae60
Address of element no. 10: 0x7fff0700ae64
```

2

2.1 Code

```
#include <stdio.h>
int main()
{
    printf("\n\n\n");
    int a = 5;
    int *b;
    b = &a;
    printf("The address of variable 'a' with value %d:\n %p", a, b);
    // & is used for reference
    // * is for de-referencing
    printf("\n\n\n");
    return 0;
}
```

2.2 Output

A screenshot of a terminal window with a dark background. It displays the output of the C program: "The address of variable 'a' with value 5:" followed by a new line and the memory address "0x7fffd4906b9c".

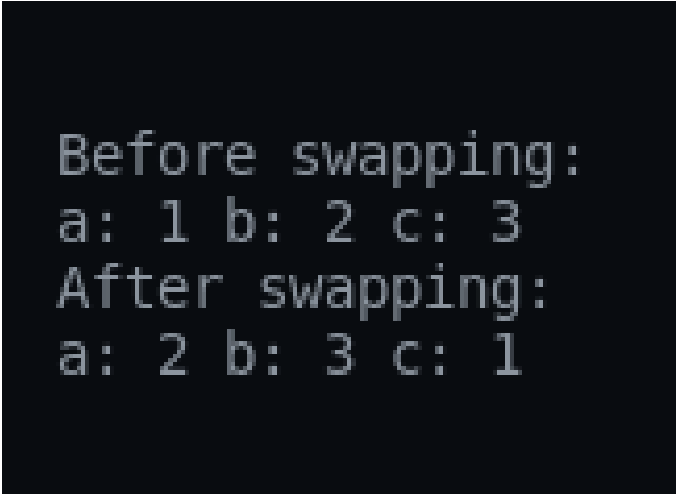
```
The address of variable 'a' with value 5:
0x7fffd4906b9c
```

3

3.1 Code

```
#include <stdio.h>
void swap(int *x, int *y, int *z)
{
    int t;
    t = *x;
    *x = *y;
    *y = *z;
    *z = t;
}
int main()
{
    printf("\n\n\n");
    int a = 1, b = 2, c = 3, t;
    printf("\n Before swapping: ");
    printf("\n a: %d b: %d c: %d", a, b, c);
    swap(&a, &b, &c);
    // call by reference causes changes in actual memory of variables
    printf("\n After swapping: ");
    printf("\n a: %d b: %d c: %d", a, b, c);
    printf("\n\n\n");
    return 0;
}
```

3.2 Output



```
Before swapping:
a: 1 b: 2 c: 3
After swapping:
a: 2 b: 3 c: 1
```

4

4.1 Code

```
#include <stdio.h>
int factorial(int *a)
{
    int t = *a - 1;
    if (*a == 1 || *a == 0)
        return 1;    // 0!=1 and 1!=1
    else
        return *a * factorial(&t); // n! = n*(n-1)!
}
int main()
{
    printf("\n\n\n");
    int n = 5;
    printf("The factorial of %d is: %d", n, factorial(&n));
    printf("\n\n\n");
    return 0;
}
```

4.2 Output

A screenshot of a terminal window with a black background and white text. The text displayed is "The factorial of 5 is: 120".

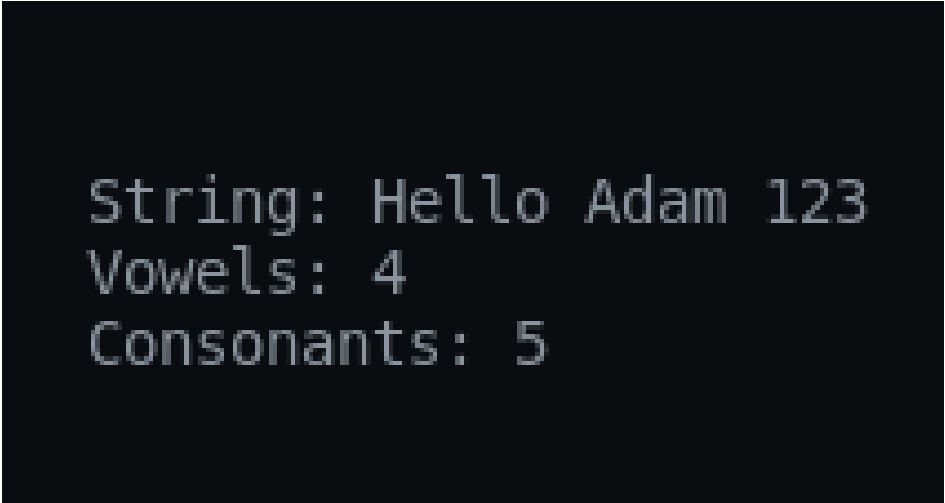
The factorial of 5 is: 120

5

5.1 Code

```
#include <stdio.h>
int main()
{
    printf("\n\n\n");
    char str[] = "Hello Adam 123";
    int vowels = 0, consonants = 0;
    char *s = str;
    for (int i = 0; s[i] != '\0';)
    {
        if (
            (*s == 'A' || *s == 'E' || *s == 'I' || *s == 'O' || *s == 'U') ||
            (*s == 'a' || *s == 'e' || *s == 'i' || *s == 'o' || *s == 'u'))
            ++vowels;      // if s[i] is vowel
        else if (
            ((*s >= 'A' && *s <= 'Z') || (*s >= 'a' && *s <= 'z')) &&
            (!(
                (*s == 'A' || *s == 'E' || *s == 'I' || *s == 'O' || *s == 'U') ||
                (*s == 'a' || *s == 'e' || *s == 'i' || *s == 'o' || *s == 'u'))))
            ++consonants;  // if s[i] is alphabet but not vowel
        *s++;
    }
    printf("\n String: %s", str);
    printf("\n Vowels: %d \n Consonants: %d", vowels, consonants);
    printf("\n\n\n");
    return 0;
}
```

5.2 Output

A screenshot of a terminal window with a black background and white text. The output shows three lines: 'String: Hello Adam 123', 'Vowels: 4', and 'Consonants: 5'.

```
String: Hello Adam 123
Vowels: 4
Consonants: 5
```

6

6.1 Code

```
#include <stdio.h>
void swap(int *a, int *b)
{
    int t;
    t = *a;
    *a = *b;
    *b = t;
}
void print_array(int a[], int n)
{
    for (int i = 0; i < n; ++i)
        printf(" %d ", a[i]);
    printf("\n");
}
int main()
{
    printf("\n\n\n");
    int a[] = {3, 2, 6, 4, 7, 1, 3, 6, 9, 0, 5};
    int n = 11, i, j;
    printf("\n The original array: \n");
    print_array(a, n);
    for (i = 0; i < n; ++i) //selection sorting
    {
        int *b = a;
        for (j = 0; j < n - 1; ++j)
        {
            if (*b > *(b + 1))
                swap(&(*b), &*(b + 1));
            *b++;
        }
    }
    printf("\n The ordered array: \n");
    print_array(a, n);
    printf("\n\n\n");
    return 0;
}
```

6.2 Output

The original array:

3 2 6 4 7 1 3 6 9 0 5

The ordered array:

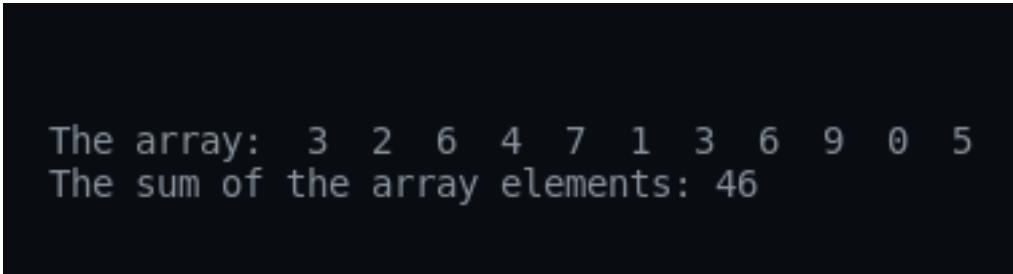
0 1 2 3 3 4 5 6 6 7 9

7

7.1 Code

```
#include <stdio.h>
int main()
{
    printf("\n\n\n");
    int a[] = {3, 2, 6, 4, 7, 1, 3, 6, 9, 0, 5};
    int n = 11,i,s=0;
    int *b=a;
    printf("\n The array: ");
    for(i=0;i<n;++i)
    {
        printf(" %d ", *b);
        s+=*b; // add elemnts to s one by one
        *b++;
    }
    printf("\n The sum of the array elements: %d",s);
    printf("\n\n\n");
    return 0;
}
```

7.2 Output

A screenshot of a terminal window with a black background and light gray text. It shows the output of the C program: "The array: 3 2 6 4 7 1 3 6 9 0 5" on the first line and "The sum of the array elements: 46" on the second line.

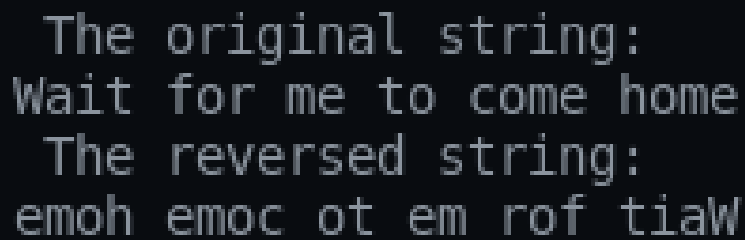
The array: 3 2 6 4 7 1 3 6 9 0 5
The sum of the array elements: 46

8

8.1 Code

```
#include <stdio.h>
int main()
{
    printf("\n\n\n");
    char str[] = "Wait for me to come home";
    char *s = str;
    int length = 0;
    for (length = 0; str[length] != '\0'; ++length);
    printf("\n The original string: \n%s", str);
    printf("\n The reversed string: \n");
    for (int i = length - 1; i >= 0; --i)
        printf("%c", *(s + i)); // printing the string from backwards
    printf("\n\n\n");
    return 0;
}
```

8.2 Output



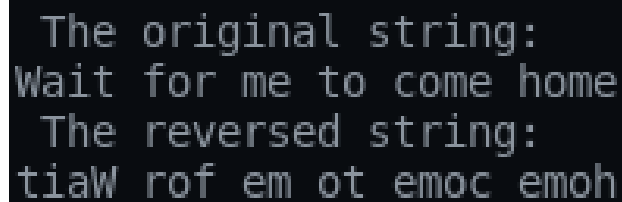
```
The original string:
Wait for me to come home
The reversed string:
emoh emoc ot em rof tiaW
```

9

9.1 Code

```
#include <stdio.h>
int main()
{
    printf("\n\n\n");
    char str[] = "Wait for me to come home";
    char *s = str;
    int length = 0, i, j = 0, k;
    printf("\n The original string: \n%s", str);
    printf("\n The reversed string: \n");
    for (i = 0; str[i] != '\0';)
    {
        for (length = j; str[length] != ' ' && str[length] != '\0'; ++length);
        //length of next word
        if (i == 0)
            i = -1;
        ++i;
        for (int k = length - 1; k >= j && str[k] != '\0'; --k, ++i)
            printf("%c", *(s + k)); //printing the from backwards
        j = length + 1;
        printf(" ");
    }
    printf("\n\n\n");
    return 0;
}
```

9.2 Output

A screenshot of a terminal window with a black background and white text. The output shows the original string "Wait for me to come home" followed by the reversed string "tiaW rof em ot emoc emoh".

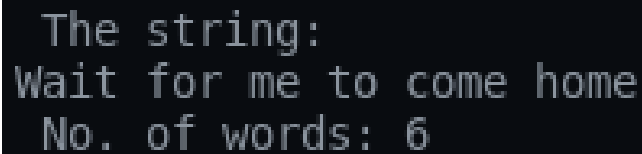
```
The original string:
Wait for me to come home
The reversed string:
tiaW rof em ot emoc emoh
```

10

10.1 Code

```
#include <stdio.h>
int main()
{
    printf("\n\n\n");
    char str[] = "Wait for me to come home";
    int i, j = 0, k = 0;
    printf("\n");
    printf("\n The string: \n%s", str);
    for (i = 0; str[i] != '\0'; ++i)
    {
        if (str[i] == ' ' || str[i + 1] == '\0')
            ++k;    //if a space or null charcater is encountered
    }
    printf("\n No. of words: %d", k);
    printf("\n\n\n");
    return 0;
}
```

10.2 Output

A screenshot of a terminal window with a black background and white text. The output of the program is displayed as follows:

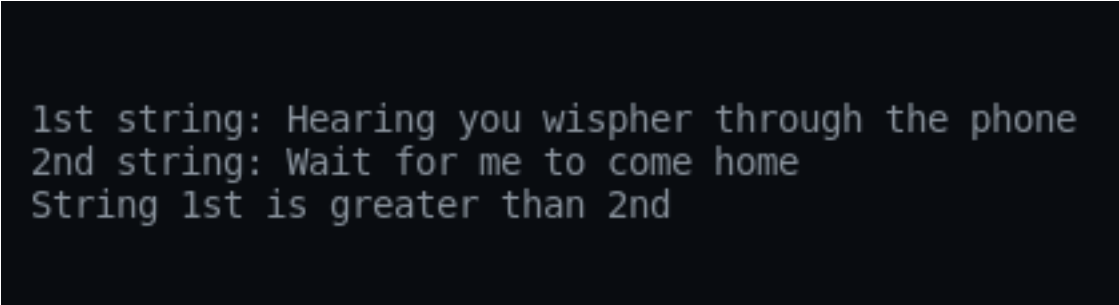
```
The string:
Wait for me to come home
No. of words: 6
```

11

11.1 Code

```
#include <stdio.h>
int main()
{
    printf("\n\n\n");
    char str1[] = "Hearing you wispher through the phone";
    char str2[] = "Wait for me to come home";
    char *s1 = str1, *s2 = str2;
    int l1 = 0, l2 = 0, i;
    for (i = 0; *s1 != '\0'; *s1++, l1++); //length of 1st string
    for (i = 0; *s2 != '\0'; *s2++, l2++); //length of 2nd string
    printf("\n 1st string: %s",str1);
    printf("\n 2nd string: %s\n ",str2);
    if (l1 > l2)
        printf("String 1st is greater than 2nd");
    else if (l1 < l2)
        printf("String 2nd is greater than 1st");
    else
        printf("Both strings have equal lengths");
    printf("\n\n\n");
    return 0;
}
```

11.2 Output

A screenshot of a terminal window with a dark background. It displays the output of the C program: three blank lines, followed by '1st string: Hearing you wispher through the phone', '2nd string: Wait for me to come home', and 'String 1st is greater than 2nd'.

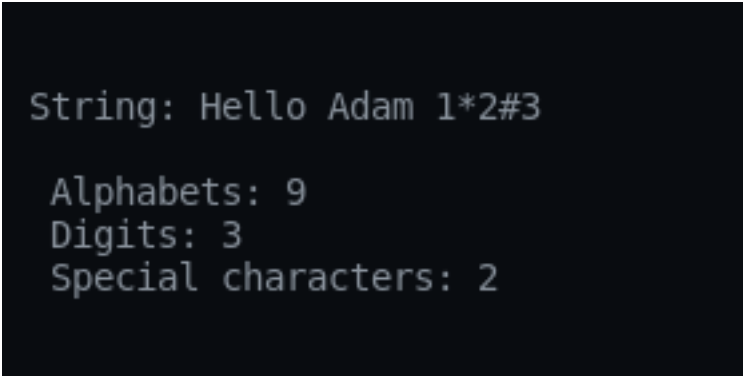
```
1st string: Hearing you wispher through the phone
2nd string: Wait for me to come home
String 1st is greater than 2nd
```

12

12.1 Code

```
#include <stdio.h>
int main()
{
    printf("\n\n\n");
    char str[] = "Hello Adam 1*2#3";
    int alphabets = 0, digits = 0, special = 0;
    char *s = str;
    for (int i = 0; s[i] != '\0';)
    {
        if ((*s >= 'A' && *s <= 'Z') || (*s >= 'a' && *s <= 'z'))
            ++alphabets;
        else if (*s >= '0' && *s <= '9')
            ++digits;
        else if (*s != ' ')
            ++special; //if charcater is not alphabet, number or space
        *s++;
    }
    printf("String: %s\n", str);
    printf("\n Alphabets: %d", alphabets);
    printf("\n Digits: %d", digits);
    printf("\n Special characters: %d", special);
    printf("\n\n\n");
    return 0;
}
```

12.2 Output

A screenshot of a terminal window with a dark background and light gray text. The output of the program is displayed as follows:

```
String: Hello Adam 1*2#3

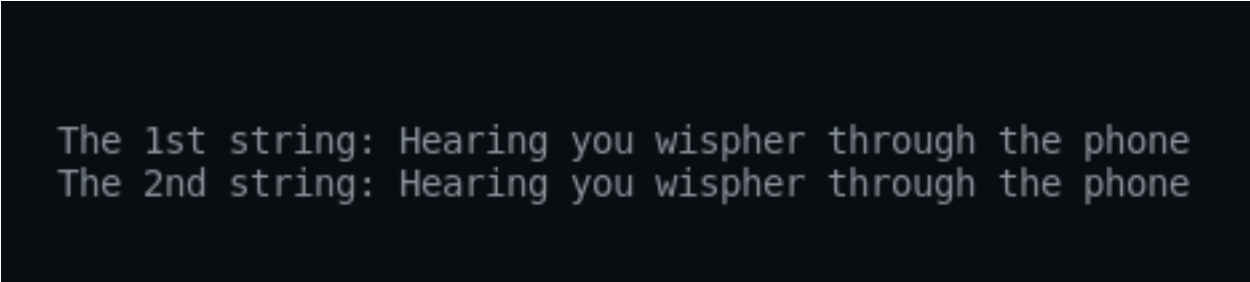
Alphabets: 9
Digits: 3
Special characters: 2
```

13

13.1 Code

```
#include <stdio.h>
int main()
{
    printf("\n\n\n");
    char str1[] = "Hearing you wispher through the phone";
    char str2[100];
    char *s1 = str1, *s2 = str2;
    for (; *s1 != '\0'; *s1++, *s2++)
        *s2 = *s1; //copying str1 in str2 by every character
    *s2 = '\0';
    printf("\n The 1st string: %s", str1);
    printf("\n The 2nd string: %s", str2);
    printf("\n\n\n");
    return 0;
}
```

13.2 Output

A screenshot of a terminal window with a dark background. It displays the output of the C program: "The 1st string: Hearing you wispher through the phone" followed by "The 2nd string: Hearing you wispher through the phone" on the next line.

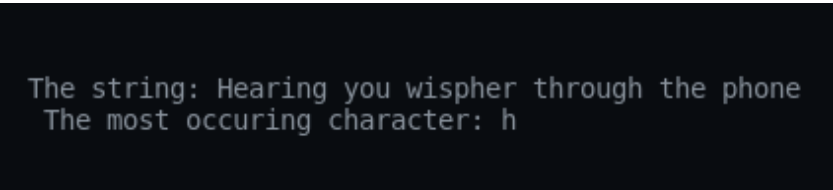
```
The 1st string: Hearing you wispher through the phone
The 2nd string: Hearing you wispher through the phone
```

14

14.1 Code

```
#include <stdio.h>
char mode(char a[], int n)
{
    int i, j, k, l = 0, m = 0;
    for (i = 0; a[i] != '\0'; ++i)
        if (a[i] >= 'A' && a[i] <= 'Z')
            a[i] += 32;    //to convert whole string in lowercase to avoid ambiguity
    for (i = 1, k = 1; i < n; ++i)
    {
        if (a[i] == a[0])
            k++; //no. of times 1st element appears
    }
    m = 0;
    for (i = 0; i < n; ++i)
    {
        l = 1;
        for (j = 0; j < n; ++j)
        {
            if ((a[i] == a[j]) && (i != j) && (a[j] != ' '))
                l++; //no. of times a[j] appears in the array
        }
        if (l > k)
            k = l, m = i; //if a[j] appears more times than a[k], replace a[k]
    }
    return a[m]; //returns mode (if any), if every number occurs once, returns first ele
}
int main()
{
    printf("\n\n\n");
    char str[] = "Hearing you wispher through the phone";
    int length;
    for (length = 0; str[length] != '\0'; ++length);
    printf("The string: %s", str);
    printf("\n The most occuring character: %c", mode(str, length));
    printf("\n\n\n");
    return 0;
}
```

14.2 Output

A screenshot of a terminal window with a dark background. It shows the output of the C program. The first line is "The string: Hearing you wispher through the phone" and the second line is "The most occuring character: h".

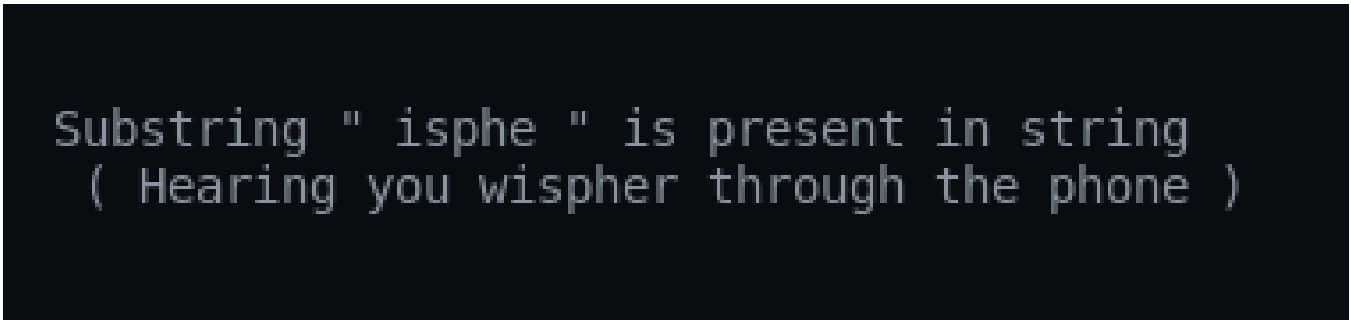
```
The string: Hearing you wispher through the phone
The most occuring character: h
```


15

15.1 Code

```
#include <stdio.h>
int main()
{
    printf("\n\n\n");
    char str[] = "Hearing you wispher through the phone";
    char check[] = "isphe";
    char *s = str, *c = check;
    int lc, i = 0, k = 0;
    for (lc = 0; check[lc] != '\0'; ++lc);
    for (; *s != '\0'; *s++)
    {
        k = 0;
        if (*s == *c)    //if 1st charcter is found
            for (i = 0; i < lc; ++i)
                if (*(s + i) == *(c + i))    //then check for all chrcaters
                    ++k;
        if (k == lc)
        {
            printf("Substring \" %s \" is present in string\n ( %s )", check, str);
            break;
        }
    }
    printf("\n\n\n");
    return 0;
}
```

15.2 Output



```
Substring " isphe " is present in string
( Hearing you wispher through the phone )
```

16

16.1 Code

```
#include <stdio.h>
#include <string.h>
int main()
{
    printf("\n\n\n");
    char str1[] = "Hearing you wispher through the phone";
    char str2[] = "Wait for me to come home", str3[100] = "Photograph";
    printf("\n The length of 1st string: %ld", strlen(str1));
    strcpy(str3, str1);
    printf("\n 3rd string after copying: %s", str3);
    strcat(str2, str1);
    printf("\n 2nd string after concatenation: %s", str2);
    int t = strcmp(str1, str2);
    printf("\n Strings 1st and 2nd are ");
    if (!t)
        printf("same");
    else
        printf("not same");
    printf("\n\n\n");
    return 0;
}
```

16.2 Output

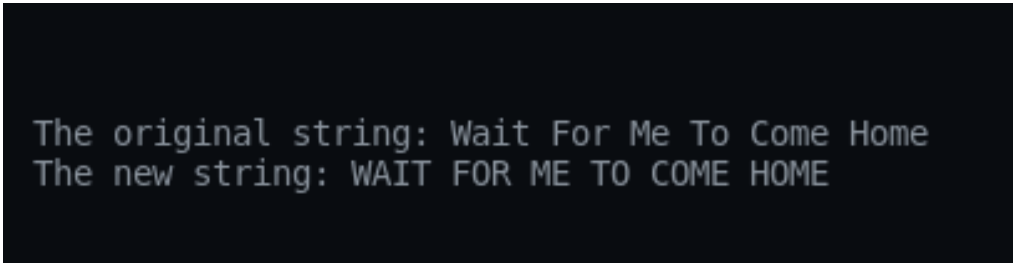
```
The length of 1st string: 37
3rd string after copying: Hearing you wispher through the phone
2nd string after concatenation: Wait for me to come homeHearing you wispher through the phone
Strings 1st and 2nd are not same
```

17

17.1 Code

```
#include <stdio.h>
int main()
{
    printf("\n\n\n");
    char str[] = "Wait For Me To Come Home";
    char *s = str;
    printf("\n The original string: %s", str);
    printf("\n The new string: ");
    for (; *s != '\0'; *s++)
    {
        if (*s >= 'a' && *s <= 'z')
            *s = *s - 32;
        //if anycharacter is lowercase, change into uppercase
        printf("%c", *s);
    }
    printf("\n\n\n");
    return 0;
}
```

17.2 Output



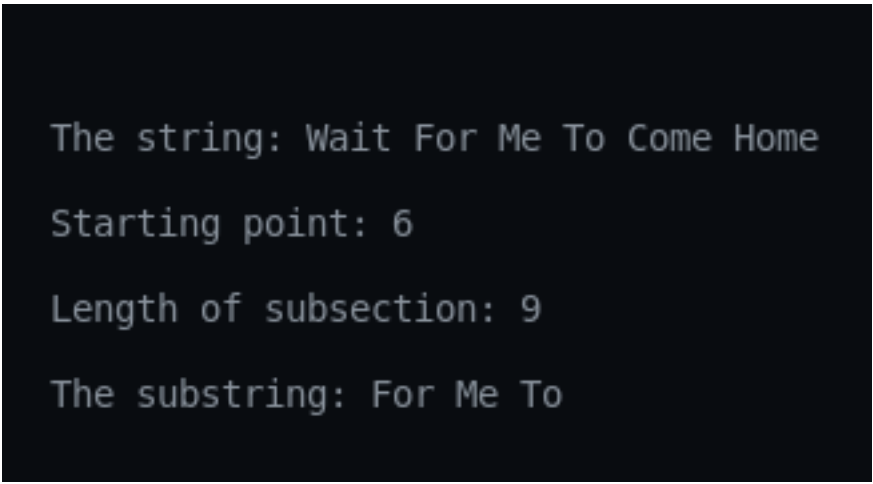
```
The original string: Wait For Me To Come Home
The new string: WAIT FOR ME TO COME HOME
```

18

18.1 Code

```
#include <stdio.h>
int main()
{
    printf("\n\n\n");
    char str[] = "Wait For Me To Come Home";
    char *s = str;
    int a, b;
    printf("\n The string: %s\n", str);
    printf("\n Starting point: ");
    scanf("%d", &a);    //from which character to trim
    printf("\n Length of subsection: ");
    scanf("%d", &b);    //length of substring
    printf("\n The substring: ");
    for (int i = 0; *s != '\0'; *s++, ++i)
    {
        if (i == a - 1)
            for (; i < a - 1 + b; ++i)
                printf("%c", *s++);
    }
    printf("\n\n\n");
    return 0;
}
```

18.2 Output

A screenshot of a terminal window with a dark background and light gray text. The output shows the program's execution results, including the original string, the starting point, the length of the subsection, and the resulting substring.

```
The string: Wait For Me To Come Home
Starting point: 6
Length of subsection: 9
The substring: For Me To
```

19

19.1 Code

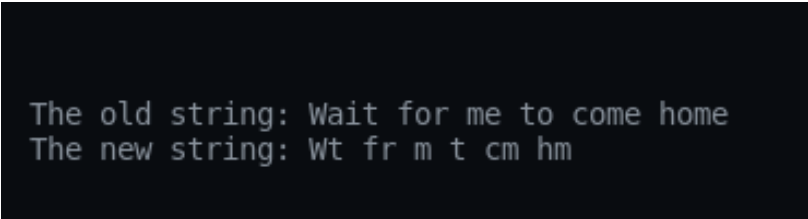
19.2 Output

20

20.1 Code

```
#include <stdio.h>
int deletion(char a[], char s, int n)
{
    int i, j, pos = -1;
    for (i = 0; i < n; ++i)
    {
        pos = -1;
        if (a[i] == s)
            pos = i;
        if (pos != -1)
        {
            for (j = pos; j < n; ++j)
                a[j] = a[j + 1];    //if any vowel found replace with next element
            a[j] = 0;
            --n, --i;
        }
    }
    return n;
}
int main()
{
    char str[] = "Wait for me to come home";
    printf("\n The old string: %s", str);
    int n = 0;
    for (n = 0; str[n] != '\0'; ++n);
    for (int i = 0; str[i] != '\0'; ++i)
    {
        if (
            (str[i] == 'A' || str[i] == 'E' || str[i] == 'I' || str[i] == 'O' ||
             str[i] == 'U') || (str[i] == 'a' || str[i] == 'e' || str[i] == 'i' ||
             str[i] == 'o' || str[i] == 'u'))
            n = deletion(str, str[i], n), --i;    //remove any vowel
    }
    printf("\n The new string: %s", str);
    return 0;
}
```

20.2 Output



```
The old string: Wait for me to come home
The new string: Wt fr m t cm hm
```