भारतीय सूचना प्रौद्योगिकी संस्थान गुवाहाटी

Indian Institute of Information Technology Guwahati

COMPUTER PROGRAMMING LAB (CS110)

SOLUTIONS–10

1. Write a program in C to print the address of each character in a given string. The string is user input.

```c
#include <stdio.h>

#define MAX_SIZE 100

int main() {
    printf("Enter a string: ");
    char a[MAX_SIZE];
    gets(a);
    int i = 0;
    for (; a[i]; i++)
        printf("Address %p contains %c (ASCII: %03d).\n", a + i, a[i], a[i]);
    printf("Address %p contains %c (ASCII: %03d).\n", a + i, a[i], a[i]);
}
```

2. Write a menu-based (use `do-while` loop and `switch-case` construct to create the menu) program in C to perform the following operations on strings:

   i. Concatenate two strings without using the `strcat` function. The strings are user inputs.

   ii. Concatenate two strings using the `strcat` function. The strings are user inputs.

   iii. Compare two strings without using the `strcmp` function. The strings are user inputs.

   iv. Compare two strings using the `strcmp` function. The strings are user inputs.

   v. Calculate the length of a string without using the `strlen` function. The string is user input.

   vi. Calculate the length of a string using the `strlen` function. The string is user input.

   vii. Copy one string to another string without using the `strcpy` function. The string is user input.

viii. Copy one string to another string using the `strcpy` function. The string is user input.

ix. A menu "exit" to terminate the program.

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

#define MAX_SIZE 100

void mystrcat1(char a[], char b[]) {
    int i = 0, j = 0;
    while (a[i])
        i++;
    while (b[j])
        a[i++] = b[j++];
    a[i] = '\0';
}

void mystrcat2(char *a, char *b) {
    while (*a++)
        ;
    a--;
    while (*b)
        *a++ = *b++;
    *a = '\0';
}

int mystrcmp(char a[], char b[]) {
    for (int i = 0; a[i] && b[i]; i++)
        if (a[i] != b[i])
            return a[i] - b[i];
    return 0;
}

int mystrlen(char a[]) {
    int i = 0;
    while (a[i])
        i++;
    return i;
}

void mystrcpy(char *a, char *b) {
    while (*b)
        *a++ = *b++;
}


int main() {
    do {
        int n = 0;
```

```
        printf(
            "1 - Concatenate two strings without using strcat.\n"
            "2 - Concatenate two strings using strcat.\n"
            "3 - Compare two strings without using strcmp.\n"
            "4 - Compare two strings using strcmp.\n"
            "5 - Calculate length of a string without using strlen.\n"
            "6 - Calculate length of a string using strlen.\n"
            "7 - Copy one string to another string without using strcpy.\n"
            "8 - Copy one string to another string using strcpy.\n"
            "9 - Exit.\n"
            "Enter your Choice: "
        );
        scanf("%d", &n);
        char a[MAX_SIZE] = "", b[MAX_SIZE] = "";
        switch (n) {
            case 1: printf("Enter the first string: ");
                    fflush(stdin);
                    gets(a);
                    fflush(stdin);
                    printf("Enter the second string: ");
                    gets(b);
                    printf("After concatenation: ");
                    //mystrcat1(a, b); // an alternative
                    mystrcat2(a, b);
                    puts(a);
                    break;
            case 2: printf("Enter the first string: ");
                    fflush(stdin);
                    gets(a);
                    fflush(stdin);
                    printf("Enter the second string: ");
                    gets(b);
                    printf("After concatenation: ");
                    strcat(a, b);
                    puts(a);
                    break;
            case 3: printf("Enter the first string: ");
                    fflush(stdin);
                    gets(a);
                    fflush(stdin);
                    printf("Enter the second string: ");
                    gets(b);
                    puts(mystrcmp(a, b) ? "Different." : "Same.");
                    break;
            case 4: printf("Enter the first string: ");
                    fflush(stdin);
                    gets(a);
                    fflush(stdin);
                    printf("Enter the second string: ");
                    gets(b);
```

```
                    puts(strcmp(a, b) ? "Different." : "Same.");
                    break;
            case 5: printf("Enter the string: ");
                    fflush(stdin);
                    gets(a);
                    printf("Length: %d\n", mystrlen(a));
                    break;
            case 6: printf("Enter the string: ");
                    fflush(stdin);
                    gets(a);
                    printf("Length: %d\n", strlen(a));
                    break;
            case 7: printf("Enter the string: ");
                    fflush(stdin);
                    gets(a);
                    mystrcpy(b, a);
                    puts(b);
                    break;
            case 8: printf("Enter the string: ");
                    fflush(stdin);
                    gets(a);
                    strcpy(b, a);
                    puts(b);
                    break;
            case 9: return 0;
        }
    } while (1);
    return 0;
}
```

3. Write a program in C to convert all lowercase characters to uppercase in a string. The string is user input.

```
#include <stdio.h>

#define MAX_SIZE 100

void toUpper(char *string) {
    while (*string)
        *string++ = (*string >= 'a' && *string <= 'z')
                  ? (*string + 'A' - 'a') : *string;
}

int main() {
    char string[MAX_SIZE] = "";
    printf("Enter a string: ");
    gets(string);
    printf("Uppercase: ");
    toUpper(string);
    puts(string);
```

```
    return 0;
}
```

4. Write a program in C to count the number of vowels in a string. The string is user input.

```c
#include <stdio.h>

#define MAX_SIZE 100

int isVowel(char c) {
    return c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u' ||
           c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U';
}

int countVowels(char *string) {
    int count = 0;
    while (*string)
        count = isVowel(*string++) ? count + 1 : count;
    return count;
}

int main() {
    char string[MAX_SIZE] = "";
    printf("Enter a string: ");
    gets(string);
    printf("The number of vowels is %d.\n", countVowels(string));
    return 0;
}
```

5. Write a program in C to check whether a string is a palindrome. The string is user input.

```c
#include <stdio.h>

#define MAX_SIZE 100

int isPalindrome(char *string) {
    int length = 0;
    while (string[length])
        length++;
    for (int i = 0, j = length - 1; i <= j; i++, j--)
        if (string[i] != string[j])
            return 0;
    return 1;
}

int main() {
    char string[MAX_SIZE] = "";
    printf("Enter a string: ");
```

```
    gets(string);
    printf(isPalindrome(string) ? "The string is palindrome." : "The string is not palindrome.");
    return 0;
}
```

6. Write a program in C to reverse a string. The string is user input.

```
#include <stdio.h>

#define MAX_SIZE 100

void swap(char *s, int i, int j) {
    char c = s[i];
    s[i] = s[j];
    s[j] = c;
}

void reverse(char *string) {
    int length = 0;
    while (string[length])
        length++;
    for (int i = 0, j = length - 1; i < j; i++, j--)
        swap(string, i, j);
}

int main() {
    char string[MAX_SIZE] = "";
    printf("Enter a string: ");
    gets(string);
    reverse(string);
    printf("Reverse string: ");
    puts(string);
    return 0;
}
```

7. Write a program in C to word-wise reverse a string. The string is user input. For instance, for an input string

   `This is an example.`

   the output should be

   `example. an is This`

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

```c
#define MAX_SIZE 100

void swap(char *s, int i, int j) {
    char c = s[i];
    s[i] = s[j];
    s[j] = c;
}

void reverse(char *string) {
    int length = strlen(string);
    char *buffer = calloc(length, sizeof (char));
    int end = length - 1, start = length - 1, k = 0;
    while (end >= 0) {
        start = end;
        while (start != -1 && string[start] != ' ' &&  string[start] != '\t')
            start--;
        for (int i = start + 1; i <= end; i++, k++)
            buffer[k] = string[i];
        buffer[k++] = string[start];
        end = start - 1;
    }
    for(int i = 0; i < length; i++)
        string[i] = buffer[i];
    free(buffer);
    buffer = NULL;
}

int main() {
    char string[MAX_SIZE] = "";
    printf("Enter a string: ");
    gets(string);
    reverse(string);
    printf("Word-wise Reverse string: ");
    puts(string);
    return 0;
}
```

8. Write a program in C to word-wise reverse each sentence in a paragraph. The paragraph (string) is user input. For instance, for an input string

   ```
   This is an example. Another sentence goes here.
   ```

   the output should be

   ```
   example an is This. here goes sentence Another.
   ```

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

#define MAX_SIZE 100

void swap(char *s, int i, int j) {
    char c = s[i];
    s[i] = s[j];
    s[j] = c;
}

void reverse(char *string) {
    int length = strlen(string);
    char *buffer = calloc(length, sizeof (char));
    int sentenceStart = 0, sentenceEnd = 0, k = 0;
    while (sentenceStart < length) {
        sentenceEnd = sentenceStart;
        while (string[sentenceEnd] != '.' && string[sentenceEnd])
            sentenceEnd++;
        int end = sentenceEnd - 1, start = 0;
        while (end >= sentenceStart) {
            start = end;
            while (
                start != sentenceStart - 1 &&
                string[start] != ' ' &&
                string[start] != '\t'
            ) start--;
            for (int i = start + 1; i <= end; i++, k++)
                buffer[k] = string[i];
            buffer[k++] = string[start];
            end = start - 1;
        }
        k--;
        while (
            string[sentenceEnd] == '.' ||
            string[sentenceEnd] == ' ' ||
            string[sentenceEnd] == '\t'
        ) buffer[k++] = string[sentenceEnd++];
        sentenceStart = sentenceEnd;
    }
    for(int i = 0; i < length; i++)
        string[i] = buffer[i];
    free(buffer);
    buffer = NULL;
}

int main() {
    char string[MAX_SIZE] = "";
    printf("Enter a string: ");
```

```
    gets(string);
    reverse(string);
    printf("Word-wise Reverse string: ");
    puts(string);
    return 0;
}
```

9. Given a non-negative integer $n \in [0, 100000000]$, write a program in C to print it in words. For instance, for the input integer $n = 19450274$, the output should be

   `One Crore Ninety Four Lakh Fifty Thousand Two Hundred Seventy Four`

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

#define MAX_SIZE 1000

void helperOfToWords(int m, char *buffer, char *level) {
    char *_1[] = {
        "", "One ", "Two ", "Three ", "Four ", "Five ", "Six ", "Seven ",
        "Eight ", "Nine ", "Ten ", "Eleven ", "Twelve ", "Thirteen ", "Fourteen ",
        "Fifteen ", "Sixteen ", "Seventeen ", "Eighteen ", "Nineteen ", "Twenty "
    };
    char *_10[] = {
        "", "Ten ", "Twenty ", "Thirty ", "Forty ", "Fifty ", "Sixty ",
        "Seventy ", "Eighty ", "Ninety "
    };
    if (m > 0) {
        if (m < 20) strcat(buffer, _1[m]);
        else {
            strcat(buffer, _10[m / 10]);
            strcat(buffer, _1 [m % 10]);
        }
        strcat(buffer, level);
    }
}

char *toWords(int n) {
    if (n < 0) return NULL;
    char buffer[MAX_SIZE] = "";
    int m = n / 10000000;
    helperOfToWords(m, buffer, "Crore ");
    m = (n % 10000000) / 100000;
    helperOfToWords(m, buffer, "Lakh ");
    m = (n % 100000) / 1000;
    helperOfToWords(m, buffer, "Thousand ");
    m = (n % 1000) / 100;
```

```
    helperOfToWords(m, buffer, "Hundred ");
    m = n % 100;
    helperOfToWords(m, buffer, "");
    if (n == 0)
        strcpy(buffer, "Zero ");
    int length = strlen(buffer);
    if (buffer[length - 1] == ' ')
        buffer[length - 1] = '\0';
    char *string = (char *) calloc(length, sizeof (char));
    strcpy(string, buffer);
    return string;
}

int main() {
    int n = 0;
    printf("Enter a positive integer: ");
    scanf("%d", &n);
    char *string = toWords(n);
    puts(string);
    free(string);
    string = NULL;
    return 0;
}
```