----------String functions----------

1

```
#include<stdio.h>
#include<string.h>
void concat(char str1[],char str2[]){
   char result[20];
   strcpy(result,str1);
   strcat(result,str2);
   printf("The concatinated string is '%s'",result);
}
int main(){
   char string1[]="Akshay";
   char string2[]="Krishna";
   concat(string1,string2);
}
```

2

```
#include <stdio.h>

int string_compare(char str1[], char str2[]) {
   int i = 0;
   while (str1[i] != '\0' && str2[i] != '\0') {
      if (str1[i] != str2[i]) {
         return 0;
      }
      i++;
   }
   if (str1[i] == '\0' && str2[i] == '\0') {
      return 1;
```

```
    }
    return 0;
}

int main() {
    char string1[] = "Akshay";
    char string2[] = "Akshay";

    int result = string_compare(string1, string2);

    if (result == 1) {
        printf("same\n");
    } else {
        printf("not same\n");
    }

    return 0;
}
```

3

```
#include<stdio.h>
#include<string.h>
int main(){
    char string[]="Hello world!";
    printf("the length of the string is %ld",strlen(string));
}
```

4

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

```
#include <string.h>

int main(){
char firstName[10];
char lastName[10];

strcpy(firstName,"Akshay");
strcpy(lastName,"krishna");

printf("Name = %s %s",firstName,lastName);
return 0;
}
```

5

```
#include<stdio.h>
#include<string.h>
int main(){
   char string1[20],string2[20];
   printf("Enter the string 1:");
   scanf("%s",string1);
   printf("Enter the string 2:");
   scanf("%s",string2);
   printf("string1+string2 = %s",strcat(string1,string2));
   return 0;
}
```

6

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

```
int main() {
    char s1[100], s2[100];
    printf("Enter string 1: ");
    scanf("%s", s1);
    printf("Enter string 2: ");
    scanf("%s", s2);

    int result = strcmp(s1, s2);

    if (result > 0) {
        printf("String 1 is bigger\n");
    } else if (result == 0) {
        printf("Both are the same\n");
    } else {
        printf("String 2 is bigger\n");
    }

    return 0;
}
```

7

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

int main() {
    char s1[100];
    printf("Enter the string: ");
```

```
    scanf("%s", s1);
    for (int i = 0; s1[i] != '\0'; i++) {
        s1[i] = toupper(s1[i]);
    }
    printf("The string converted to uppercase is: %s\n", s1);
    return 0;
}
```

8

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

int main() {
    char s1[100];
    printf("Enter the string: ");
    scanf("%s", s1);
    for (int i = 0; s1[i] != '\0'; i++) {
        s1[i] = tolower(s1[i]);
    }
    printf("The string converted to lowercase is: %s\n", s1);
    return 0;
}
```

9

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

int main() {
```

```
    char s1[20], s2[20];

    printf("Enter the string: ");
    scanf("%[^\n]", s1);

    printf("Enter the substring to search: ");
    scanf("%s", s2);
    char *result = strstr(s1, s2);
    if (result != NULL) {
        printf("Substring is present in the string\n");
    } else {
        printf("Substring is not present in the string\n");
    }

    return 0;
}
```

10

```
#include<stdio.h>
#include<string.h>
int main(){
    char str[20];
    char ch;
    printf("enter a string :");
    scanf("%s",str);
    getchar();
    printf("enter a character :");
    scanf("%c",&ch);
    char *result=strchr(str,ch);
```

```
    if(result!=NULL){
        printf("charcter present in the string");
    }else{
        printf("charcter not present in the string");
    }
    return 0;
}
```

11

```
#include <stdio.h>
#include <string.h>
void reverseString(char str[]) {
    int len=strlen(str);
    int start=0;
    int end=len-1;
    while(start<end){
        char temp=str[start];
        str[start]=str[end];
        str[end]=temp;
        start++;
        end--;
    }
}

int main() {
    char str1[20];

    printf("Enter the string: ");
    scanf("%s", str1);
```

```
    reverseString(str1);

    printf("Reversed string: %s\n", str1);

    return 0;
}
```

12

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

int main() {
    char str[200];
    printf("Enter a sentence: ");
    fgets(str, sizeof(str), stdin);
    str[strcspn(str, "\n")] = '\0';

    int tokenCount = 0;

    char *token = strtok(str, " ");
    while (token != NULL) {
        tokenCount++;
        token = strtok(NULL, " ");
    }

    printf("Number of words: %d\n", tokenCount);
    return 0;
}
```

13

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h> // For strdup()

int main() {
    char str[100];

    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    str[strcspn(str, "\n")] = '\0';

    char *duplicatedStr = strdup(str);

    if (duplicatedStr == NULL) {
        printf("Memory allocation failed.\n");
        return 1;
    }
    printf("Original string: %s\n", str);
    printf("Duplicated string: %s\n", duplicatedStr);
    free(duplicatedStr);

    return 0;
}
```

14

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

```
int main() {
    char buffer[100];
    printf("Enter a string (scanf): ");
    scanf("%99s", buffer);
    printf("You entered: %s\n", buffer);
    int ch;
    while ((ch = getchar()) != '\n' && ch != EOF);
    printf("Enter a string (fgets): ");
    fgets(buffer, sizeof(buffer), stdin);
    printf("You entered: %s", buffer);

    return 0;
}
```

15

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

void trimWhitespace(char *str) {
    char *start = str;
    char *end = strlen(str) - 1;
    while (*start && isspace((unsigned char)*start)) {
        start++;
    }
    if (*start == '\0') {
        *str = '\0';
        return;
    }
```

```
    while (end > start && isspace((unsigned char)*end)) {
        end--;
    }
    *(end + 1) = '\0';
    memmove(str, start, strlen(start) + 1);
}

int main() {
    char input[100];

    printf("Enter a string with extra spaces: ");
    fgets(input, sizeof(input), stdin);
    size_t len = strlen(input);
    if (len > 0 && input[len - 1] == '\n') {
        input[len - 1] = '\0';
    }

    trimWhitespace(input);
    printf("Trimmed string: '%s'\n", input);

    return 0;
}
```

16

```
#include <stdio.h>

int findLastOccurrence(const char *str, char ch) {
    int index = -1;
    for (int i = 0; str[i] != '\0'; i++) {
```

```
        if (str[i] == ch) {
            index = i;
        }
    }

    return index;
}

int main() {
    char str[100], ch;
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);

    printf("Enter a character to search for: ");
    scanf("%c", &ch);
    int index = findLastOccurrence(str, ch);

    if (index != -1) {
        printf("Last occurrence of '%c' is at index: %d\n", ch, index);
    } else {
        printf("Character '%c' not found in the string.\n", ch);
    }

    return 0;
}
```

17

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

```
#include<stdlib.h>
#include<ctype.h>
int vowelcount(char *str){
   int count=0;
   for(int i=0;str[i]!='\0';i++){
      char ch=tolower(str[i]);
      if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
         count++;
      }
   }
   return count;
   }

int main(){
   char string[100];
   printf("enter the string :");
   fgets(string,sizeof(string),stdin);
   string[strcspn(string, "\n")] = '\0';
   int vc=vowelcount(string);
   printf("the number of vowels is %d",vc);
   return 0;
}
```

18

```
#include <stdio.h>
#include <ctype.h>
int countCharacterOccurrences(const char *str, char ch, int caseSensitive) {
   int count = 0;
   for (int i = 0; str[i] != '\0'; i++) {
```

```
        char currentChar = str[i];
        if (!caseSensitive) {
            currentChar = tolower(currentChar);
            ch = tolower(ch);
        }
        if (currentChar == ch) {
            count++;
        }
    }

    return count;
}

int main() {
    char str[100], ch;
    int caseSensitiveOption, occurrences;
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    printf("Enter the character to count: ");
    scanf("%c", &ch);
    printf("Do you want case-sensitive counting? (1 for Yes, 0 for No): ");
    scanf("%d", &caseSensitiveOption);
    occurrences = countCharacterOccurrences(str, ch, caseSensitiveOption);
    printf("The character '%c' appeared %d times in the string.\n", ch, occurrences);

    return 0;
}
```

```
#include <stdio.h>

void removeCharacter(char *str, char ch) {
    int i = 0, j = 0;
    while (str[i] != '\0') {
        if (str[i] != ch) {
            str[j] = str[i];
            j+=1;
        }
        i++;
    }
    str[j] = '\0';
}

int main() {
    char str[100], ch;

    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);

    printf("Enter the character to remove: ");
    scanf("%c", &ch);

    removeCharacter(str, ch);

    printf("Modified string: %s\n", str);

    return 0;
}
```

20

```
#include <stdio.h>
#include <ctype.h>
#include <string.h>
int isPalindrome(char *str){
   int left = 0, right = strlen(str) - 1;

   while (left < right) {
      while (left < right && !isalnum(str[left])) {
         left++;
      }
      while (left < right && !isalnum(str[right])) {
         right--;
      }

      if (tolower(str[left]) != tolower(str[right])) {
         return 0;
      }

      left++;
      right--;
   }

   return 1;
}
int main() {
   char str[100];
```

```
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    str[strcspn(str, "\n")] = '\0';

    if (isPalindrome(str)) {
        printf("The string is a palindrome.\n");
    } else {
        printf("The string is not a palindrome.\n");
    }

    return 0;
}
```

21

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

void extractSubstring(char *str, int start, int length) {
    int strLength = strlen(str);

    char substring[length + 1];

    for (int i = 0; i < length; i++) {
        substring[i] = str[start + i];
    }

    substring[length] = '\0';  // Null-terminate the substring
    printf("Extracted substring: '%s'\n", substring);
}
```

```
int main() {
    char str[100];
    int start, length;

    printf("Enter the main string: ");
    fgets(str, sizeof(str), stdin);

    // Remove newline character from the end of the string if fgets adds it
    str[strcspn(str, "\n")] = '\0';

    printf("Enter the start index: ");
    scanf("%d", &start);

    printf("Enter the length of the substring: ");
    scanf("%d", &length);

    extractSubstring(str, start, length);

    return 0;
}
```

22

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

void sortString(char *str) {
    int length = strlen(str);
    for (int i = 0; i < length - 1; i++) {
```

```
        for (int j = i + 1; j < length; j++) {
            if (str[i] > str[j]) {
                char temp = str[i];
                str[i] = str[j];
                str[j] = temp;
            }
        }
    }
}

int main() {
    char str[100];

    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);

    str[strcspn(str, "\n")] = '\0';

    sortString(str);

    printf("Sorted string: %s\n", str);

    return 0;
}
```

23

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

```
int countWords(char *str) {
    int count = 0;
    char *token = strtok(str, " ");

    while (token != NULL) {
        count++;
        token = strtok(NULL, " ");
    }

    return count;
}

int main() {
    char str[100];

    printf("Enter a sentence: ");
    fgets(str, sizeof(str), stdin);

    str[strcspn(str, "\n")] = '\0';

    int wordCount = countWords(str);

    printf("Number of words: %d\n", wordCount);

    return 0;
}
```

24

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

```
#include <string.h>

void removeDuplicates(char *str) {
   int length = strlen(str);
   int index = 0;
   int found;

   for (int i = 0; i < length; i++) {
      found = 0;

      for (int j = 0; j < i; j++) {
         if (str[i] == str[j]) {
            found = 1;
            break;
         }
      }

      if (found==1) {
         str[index++] = str[i];
      }
   }

   str[index] = '\0';
}

int main() {
   char str[100];

   printf("Enter a string: ");
```

```
    fgets(str, sizeof(str), stdin);

    str[strcspn(str, "\n")] = '\0';

    removeDuplicates(str);

    printf("Modified string without duplicates: %s\n", str);

    return 0;
}
```

25

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

char firstNonRepeatingChar(char *str) {
    int count[256] = {0};

    for (int i = 0; str[i] != '\0'; i++) {
        count[(unsigned char)str[i]]++;
    }

    for (int i = 0; str[i] != '\0'; i++) {
        if (count[(unsigned char)str[i]] == 1) {
            return str[i];
        }
    }

    return '\0';
```

```
}

int main() {
    char str[100];

    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);

    str[strcspn(str, "\n")] = '\0';

    char result = firstNonRepeatingChar(str);

    if (result == '\0') {
        printf("All characters are repeating.\n");
    } else {
        printf("First non-repeating character: %c\n", result);
    }

    return 0;
}
```

26

```
#include <stdio.h>
#include <ctype.h>

int stringToInt(char *str) {
    int result = 0;
    int i = 0;
    int negative = 0;
```

```
    if (str[i] == '-') {
        negative = 1;
        i++;
    }

    for (; str[i] != '\0'; i++) {
        if (!isdigit(str[i])) {
            printf("Error: Invalid input. Not a valid number.\n");
            return -1;
        }
        result = result * 10 + (str[i] - '0');
    }

    if (negative) {
        result = -result;
    }

    return result;
}

int main() {
    char str[100];

    printf("Enter a numeric string: ");
    fgets(str, sizeof(str), stdin);

    str[strcspn(str, "\n")] = '\0';
```

```
    int result = stringToInt(str);

    if (result != -1) {
        printf("Converted integer value: %d\n", result);
    }

    return 0;
}
```

27

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

void sortString(char *str) {
    int length = strlen(str);
    for (int i = 0; i < length - 1; i++) {
        for (int j = i + 1; j < length; j++) {
            if (str[i] > str[j]) {
                char temp = str[i];
                str[i] = str[j];
                str[j] = temp;
            }
        }
    }
}

int areAnagrams(char *str1, char *str2) {
    if (strlen(str1) != strlen(str2)) {
```

```
        return 0;
    }

    sortString(str1);
    sortString(str2);

    return strcmp(str1, str2) == 0;
}

int main() {
    char str1[100], str2[100];

    printf("Enter first string: ");
    fgets(str1, sizeof(str1), stdin);
    str1[strcspn(str1, "\n")] = '\0'; // Remove newline

    printf("Enter second string: ");
    fgets(str2, sizeof(str2), stdin);
    str2[strcspn(str2, "\n")] = '\0'; // Remove newline

    if (areAnagrams(str1, str2)) {
        printf("The strings are anagrams.\n");
    } else {
        printf("The strings are not anagrams.\n");
    }

    return 0;
}
```

28

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

void mergeStringsAlternately(char *str1, char *str2) {
    int i = 0, j = 0;
    while (str1[i] != '\0' || str2[j] != '\0') {
        if (str1[i] != '\0') {
            printf("%c", str1[i++]);
        }
        if (str2[j] != '\0') {
            printf("%c", str2[j++]);
        }
    }
    printf("\n");
}

int main() {
    char str1[100], str2[100];

    printf("Enter first string: ");
    fgets(str1, sizeof(str1), stdin);
    str1[strcspn(str1, "\n")] = '\0';

    printf("Enter second string: ");
    fgets(str2, sizeof(str2), stdin);
    str2[strcspn(str2, "\n")] = '\0';

    printf("Merged string: ");
```

```
    mergeStringsAlternately(str1, str2);

    return 0;
}
```

29

```
#include <stdio.h>
#include <ctype.h>

int countConsonants(char *str) {
    int count = 0;
    for (int i = 0; str[i] != '\0'; i++) {
        char ch = tolower(str[i]);
        if (isalpha(ch) && ch != 'a' && ch != 'e' && ch != 'i' && ch != 'o' && ch != 'u') {
            count++;
        }
    }
    return count;
}

int main() {
    char str[100];

    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    str[strcspn(str, "\n")] = '\0';

    int result = countConsonants(str);
    printf("Number of consonants: %d\n", result);
```

```
    return 0;
}
```

30

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

void replaceSubstring(char *str, const char *target, const char *replacement) {
    char buffer[1000];
    int i = 0, j = 0;
    int targetLen = strlen(target);
    int replacementLen = strlen(replacement);

    while (str[i] != '\0') {
        if (strncmp(&str[i], target, targetLen) == 0) {
            strcpy(&buffer[j], replacement);
            j += replacementLen;
            i += targetLen;
        } else {
            buffer[j++] = str[i++];
        }
    }
    buffer[j] = '\0';
    strcpy(str, buffer);
}

int main() {
    char str[1000];
```

```
    char target[100], replacement[100];

    printf("Enter main string: ");
    fgets(str, sizeof(str), stdin);
    str[strcspn(str, "\n")] = '\0';

    printf("Enter target substring: ");
    fgets(target, sizeof(target), stdin);
    target[strcspn(target, "\n")] = '\0';

    printf("Enter replacement substring: ");
    fgets(replacement, sizeof(replacement), stdin);
    replacement[strcspn(replacement, "\n")] = '\0';

    replaceSubstring(str, target, replacement);
    printf("Modified string: %s\n", str);

    return 0;
}
```

31

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

int countOccurrences(char *str, const char *sub) {
    int count = 0;
    int subLen = strlen(sub);

    for (int i = 0; str[i] != '\0'; i++) {
```

```
        if (strncmp(&str[i], sub, subLen) == 0) {
            count++;
            i += subLen - 1;
        }
    }

    return count;
}

int main() {
    char str[1000], sub[100];

    printf("Enter main string: ");
    fgets(str, sizeof(str), stdin);
    str[strcspn(str, "\n")] = '\0';

    printf("Enter substring: ");
    fgets(sub, sizeof(sub), stdin);
    sub[strcspn(sub, "\n")] = '\0';

    int result = countOccurrences(str, sub);
    printf("Number of occurrences: %d\n", result);

    return 0;
}
```

32

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

```
int customStrLen(char *str) {
    int length = 0;
    while (str[length] != '\0') {
        length++;
    }
    return length;
}

int main() {
    char str[100];

    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    str[strcspn(str, "\n")] = '\0';

    int length = customStrLen(str);
    printf("Length of the string: %d\n", length);

    return 0;
}
```