Data Structures Lab

Experiment 3:
String Operations

Project By:

Mohammed Rabeeh
Roll No: 35
TVE18CS036

Contents

1 Aim

To implement various string operations in C program and verify their output with that of standard library functions.

2 Problem Description

The following string operations has to be implemented in C language without using library functions.

1. **String Length** returns the length of the string.
2. **String Concatenation** joins two strings.
3. **String Copy** copies one string to another.
4. **String Compare** compares two strings character by character. If the first character of two strings are equal, next character of two strings are compared. This continues until the corresponding characters of two strings are different or a null character is reached.
5. Search and replace a sub-string with another.
6. Count the number of vowels, consonants and words in the given string.

3 Algorithm

3.1 String Length

1. Take input string.
2. Iterate thorough each character(ch) of string.
3. while ch != 0, count++.
4. return count.

3.2 String Concatenation

1. Take input strings, string1 and string2.
2. Let the pointer x point to string1.
3. Iterate the pointer string1 through the string until string1 = 0.
4. Copy each character of string2 to string1 while increasing string1 and string2 by 1.
5. return character pointer x.

3.3 String Copy

1. Take input string, string1.
2. Copy each character of string1 to new string, string2.
3. return string2.

3.4 String Compare

1. Take input strings, string1 and string2
2. while(*string1 and (*string1 == *string2)), increment string1 and string2
3. return string1-string2.

3.5 Sub-string Search and Replace

1. Take input strings, string, search, replace
2. Count the number of occurrences of search in string.
3. If count == 0, return string.
4. else, create a new string, result with the size:

$$size = len(string) + count * (len(search) - len(replace)) + 1$$

5. Copy original string to new string with the replacement word.
6. return new string.

3.6 Count Vowels, Consonants and Words

1. Let vcount, ccount, wcount = 0
2. Take input string.
3. Iterate through each character(ch) in the string.
4. if ch is vowel, vcount++.
5. if ch is consonant, ccount++.
6. if ch is ' ' and previous ch != ' ', wcount++.
7. return vcount, ccount, wcount+1.

4 Program Code

```
#include<stdio.h>
#define MAX_SIZE 100

int isVowel(char ch) {
   switch(ch) {
      case 'a':
      case 'e':
      case 'i':
      case 'o':
      case 'u':
      case 'A':
      case 'E':
      case 'I':
      case 'O':
      case 'U':
         return 1;
      default:
         return 0;
   }
}

int stringlen(char *s1) {
```

```
    int i = 0;
    while(*s1++)
        i++;
    return i;
}

char * stringcpy(char *s2, char *s1) {
    while((*s2 = *s1)) {
        s1++;
        s2++;
    }
    return s2;
}

int stringcmp(char *s1, char *s2) {
    while(*s1 && (*s1 == *s2)) {
        s1++;
        s2++;
    }
    return *s1-*s2;
}

char * stringcat(char *s1, char *s2) {
    char *rtrn = s1;
    while(*s1)
        s1++;
    while((*s1++ = *s2++));
    *s1 = '\0';
    return rtrn;
}

int countVowels(char s[]) {
    int i, count = 0;
    char ch;
    for(i = 0; i < stringlen(s); i++) {
        ch = s[i];
        if(isVowel(ch))
            count++;
    }
    return count;
```

```
}

int countConsonants(char s[]) {
    int i, count = 0;
    char ch;
    for(i = 0; i < stringlen(s); i++) {
        ch = s[i];
        if(!isVowel(ch) && ((ch >= 'a' && ch <= 'z') || (ch >= 'A'
            && ch <= 'Z')))
            count++;
    }
    return count;
}

int countWords(char s[]) {
   int i, count = 0;
   char ch, prevch = s[0];
   for(i = 0; i < stringlen(s); i++) {
      ch = s[i];
      if(ch == ' ' && prevch != ' ') {
         count++;
      }
      prevch = ch;
   }
   return count+1;
}

void stringreplace(char *s, char *oldw, char *neww) {
   char *result;
   int i, count = 0;
   int newwlen = strlen(neww);
   int oldwlen = strlen(oldw);
   for(i = 0; s[i] < '\0'; i++) {
      if(strstr(&s[i], oldw) == &s[i]) {
         count++;
         i += oldwlen - 1;
      }
   }
   result = (char *)malloc(i + count * (newwlen - oldwlen) + 1);
```

```
   i = 0;
   while(*s) {
      if(strstr(s, oldw) == s) {
         strcpy(&result[i], neww);
         i += newwlen;
         s += oldwlen;
      } else {
         result[i++] = *s++;
      }
   }
   result[i] = '\0';
   printf("%s\n", result);
}

void main() {
    int option;
    char string1[MAX_SIZE], string2[MAX_SIZE], replace[MAX_SIZE];
    printf("String Functions\n\n");
    printf("1. String length\n2. String concatenation\n3. String
        copy\n4. String compare\n5. Substring search and
        replace\n6. Count vowels, consonants and words\nChoose an
        option: ");
    scanf("%d", &option);
    printf("\n");
    switch(option) {
        case 1:
            printf("Enter the string: ");
            scanf(" %[^\n]", string1);
            printf("String length is %d.\n", stringlen(string1));
            break;
        case 3:
            printf("Enter string 1: ");
            scanf(" %[^\n]", string1);
            stringcpy(string2, string1);
            printf("String has been copied. New string is %s\n",
                string2);
            break;
        case 4:
            printf("Enter string 1: ");
            scanf(" %[^\n]", string1);
```

```
            printf("Enter string 2: ");
            scanf(" %[^\n]", string2);
            printf("String compare: %d\n", stringcmp(string1,
                string2));
            break;
        case 2:
            printf("Enter string 1: ");
            scanf(" %[^\n]", string1);
            printf("Enter string 2: ");
            scanf(" %[^\n]", string2);
            stringcat(string1, string2);
            printf("String concatenation: %s\n", string1);
            break;
      case 5:
          printf("Enter string: ");
          scanf(" %[^\n]", string1);
          printf("Enter substring to search: ");
          scanf(" %[^\n]", string2);
          printf("Enter replace string: ");
          scanf(" %[^\n]", replace);
          stringreplace(string1, string2, replace);
          break;
      case 6:
            printf("Enter string: ");
            scanf(" %[^\n]", string1);
          printf("No. of Vowels: %d\n", countVowels(string1));
          printf("No. of Consonants: %d\n",
              countConsonants(string1));
          printf("No. of Words: %d\n", countWords(string1));
            break;
        default:
            printf("Wrong option! Try again.");
    }
}
```

5 Output

```
String Functions

1. String length
2. String concatenation
3. String copy
4. String compare
5. Substring search and replace
6. Count vowels, consonants and words
Choose an option: 2

Enter string 1: Rabeeh
Enter string 2: is Batman
String concatenation: Rabeeh is Batman
rabeehrz@BatPC:~/college/s3dslab/cycle1$ ./a.out
String Functions

1. String length
2. String concatenation
3. String copy
4. String compare
5. Substring search and replace
6. Count vowels, consonants and words
Choose an option: 5

Enter string: Today is monday.
Enter substring to search: monday
Enter replace string: tuesday
Today is tuesday.
rabeehrz@BatPC:~/college/s3dslab/cycle1$ ./a.out
String Functions

1. String length
2. String concatenation
3. String copy
4. String compare
5. Substring search and replace
6. Count vowels, consonants and words
Choose an option: 6

Enter string: My name is Rabeeh.
No. of Vowels: 6
No. of Consonants: 8
No. of Words: 4
```

6 Result

The string operations were implemented in C language and the output waas verified with that of standard library functions.