PROGRAMACIÓN ESTRUCTURADA

PROF. PEDRO NÚÑEZ YEPIZ

REPORTE DE PRACTICA #7
CADENAS

KEVIN ALEJANDRO GONZALEZ TORRES
372354 – GPO 932

INTRODUCCIÓN

En este reporte se mostrara el uso de cadenas, de sus distintos usos en diversas aplicaciones diarias y de como funcionan las funciones de la principal libreria que trabaja con cadenas <string.h>

COMPETENCIA

Se practicará el uso de las cadenas.

FUNDAMENTOS

Strings:

https://www.geeksforgeeks.org/strings-in-c/

https://www.programiz.com/c-programming/c-strings

PROCEDIMIENTO

1.- Función Que reciba como parámetro una cadena y la convierta a MAYUSCULAS

```
102
103  char *uppercaseString(char string[])
104  {
105      int i;
106
107      for (i = 0; string[i] != '\0'; i++)
108      {
109          if (string[i] >= 'a' && string[i] <= 'z')
110          {
111              string[i] -= 32;
112          }
113      }
114
115      return (string);
116  }
117
```

2.- Función Que reciba como parámetro una cadena y REVIERTA la cadena

```
118  char *reversedString(char string[])
119  {
120      int length = 0, reverse = 0, i = 0;
121      char temp = 0;
122
123      while (string[length++] != '\0')
124          ;
125
126      length--;
127      length--;
128
129      while (reverse < length)
130      {
131          temp = string[reverse];
132          string[reverse] = string[length];
133          string[length] = temp;
134          reverse++;
135          length--;
136      }
137      uppercaseString(string);
138      return (string);
139  }
```

3.- Función que reciba como parámetro una cadena y la imprima LETRA POR LETRA

```
141 void lettbylettString(char string[])
142   {
143       int i;
144
145       printf("Entered string letter by letter is:");
146
147       uppercaseString(string);
148
149       for (i = 0; string[i] != '\0'; i++)
150       {
151           printf("%c\n", string[i]);
152       }
153   }
```

4.- Función que reciba como parámetro una cadena y la imprima LETRA POR LETRA y REVERTIDA

```
155 void lettbylettReverseString(char string[])
156 {
157     int i = 0;
158     printf("Entered reverse string letter by letter is:");
159
160     reversedString(string);
161     uppercaseString(string);
162     for (i = 0; string[i] != '\0'; i++)
163     {
164         printf("%c\n", string[i]);
165     }
166 }
```

5.- Función que reciba como parámetro una cadena y la imprima en FORMA DE ESCALERA empezando por la DERECHA

```c
void ladderString(char string[])
{
    int length = 0;

    uppercaseString(string);

    printf("The string in \"stair\" order is: ");
    while (string[length++] != '\0')
        ;

    for (int i = 0; i < length; i++)
    {
        for (int j = 0; j < length - i; j++)
        {
            printf("%c", string[j]);
        }
        printf("\n");
    }
}
```

6.- Función que reciba como parámetro una cadena y la imprima en FORMA DE ESCALERA y REVERTIDA empezando por la DERECHA

```c
> void ladderString(char string[])···

void ladderReverseString(char string[])
{
    int length = 0;

    uppercaseString(string);
    reversedString(string);

    printf("The reverse string in \"stair\" order is: ");
    while (string[length++] != '\0')
        ;

    for (int i = 0; i < length; i++)
    {
        for (int j = 0; j < length - i; j++)
        {
            printf("%c", string[j]);
        }
        printf("\n");
    }
}
```

7.- Función que reciba como parámetro una cadena y la imprima en FORMA DE ESCALERA empezando por la IZQUIERDA

```c
void ladderString_2(char string[])
{
    int i, j;
    int length = 0;
    uppercaseString(string);

    while (string[length++] != '\0')
        ;
    printf("The string in \"stair\" order is: ");
    puts(string);
    while (length > 1)
    {
        for (int i = 0; i < length - 1; i++)
        {
            string[i] = string[i + 1];
        }
        length--;
        string[length] = '\0';
        printf("%s\n", string);
    }
}
```

8.- Función que reciba como parámetro una cadena y la imprima en FORMA DE ESCALERA y REVERTIDA empezando por la IZQUIERDA

```c
void ladderReverseString_2(char string[])
{
    int i, j;
    int length = 0;
    uppercaseString(string);
    reversedString(string);

    while (string[length++] != '\0')
        ;
    printf("The reverse string in \"stair\" order is: ");
    puts(string);
    while (length > 1)
    {
        for (int i = 0; i < length - 1; i++)
        {
            string[i] = string[i + 1];
        }
        length--;
        string[length] = '\0';
        printf("%s\n", string);
    }
}
```

9.- Función que reciba como parámetro una cadena e imprima solo las CONSONANTES

```
254 void consonantString(char string[])
255 {
256     int i;
257 
258     uppercaseString(string);
259     for (i = 0; string[i] != '\0'; i++)
260     {
261         char character = string[i];
262 
263         if ((character >= 'A' && character <= 'Z') && (character != 'A' && character != 'E' && character != 'I' && character != 'O' && character != 'U'))
264         {
265             printf("%c", string[i]);
266         }
267     }
268 }
```

10.- Función que reciba como parámetro una cadena e imprima solo las VOCALES

```
270 void vocalString(char string[])
271 {
272     int i;
273 
274     uppercaseString(string);
275     for (i = 0; string[i] != '\0'; i++)
276     {
277         char character = string[i];
278 
279         if ((character >= 'A' && character <= 'Z') && (character == 'A' || character == 'E' || character == 'I' || character == 'O' || character == 'U'))
280         {
281             printf("%c", string[i]);
282         }
283     }
284 }
```