Universidad del Bío-Bío

Tarea 3: “Sopa de Letras”

Integrantes: -Diego Jiménez Muñoz

-Andrés Nieto

Profesor: Christian Vidal

Carrera: Ingeniería Civil en Informática

Fecha: 7 de agosto del 2019

Propuesta Enunciado Tarea 3

Crear un programa en C que permita jugar el juego "*sopa de letras*". Requisitos de la aplicación:

- Deberá generar una matriz de caracteres de **n x n**.
- Deberá permitir el ingreso de **k palabras** por teclado, de tamaño igual o menor a n, las cuales se ubicarán en la matriz en una posición al azar en una fila, columna, diagonal principal o diagonal secundaria.
- Un carácter puede pertenecer a lo más a dos palabras
- Luego del ingreso de las palabras a la matriz, deberá llenar el resto de la matriz con caracteres del alfabeto en forma aleatoria.
- Una vez terminada la generación de la matriz, esta deberá ser mostrada por pantalla.
- El usuario deberá encontrar las palabras. Por cada palabra encontrada, el jugador deberá digitarla y su programa deberá verificar que efectivamente la palabra se encuentra presente, mostrando en pantalla la coordenada (celda y columna) en donde comienza la palabra.
- Su programa deberá permitir un máximo de 3 errores.
- El programa debe registrar el tiempo que demora el usuario en encontrar las k palabras ingresadas.

Entradas:

n: tamaño de la matriz nxn.

k: numero de palabras a ingresar.

palabras[30]: arreglo de tipo char para ir ingresando palabras.

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

int aleatorio(int n, int largo);                //declaración de funciones
int ncaracteres(char cadena[30]);
void horizontal(char sopa[100][100],char palabras[30],int largo,int x, int y,int *repetir);
void vertical(char sopa[100][100],char palabras[30],int largo,int x, int y,int *repetir);
void diagonalp(char sopa[100][100],char palabras[30],int largo,int x, int y,int *repetir);
void diagonals(char sopa[100][100],char palabras[30],int largo,int x, int y,int *repetir,int n);
int revisar(char mpalabras[100][100], char palabras[30], int largo, int k,int *num);

main()
{
int n,k;              //declaración de variables
int i,j,p,q,a,c;
int x,y;
int largo;
int repetir;
int cont=0;
int respuesta=0;
int direccion;
int num=0;
int ejex[100];
int ejey[100];
int casillas[100][100];
char sopa[100][100];
char mpalabras[100][100];
char palabras[30];
char b='a';

srand(time(NULL));
printf("*****BIENVENID@ AL JUEGO DE LA SOPA DE LETRAS*****\n\nPara comenzar es necesario ingresar algunos datos...\n\n");
printf("1-Introduzca el tamanho de la sopa(matriz simetrica): ");  //tamaño de la matriz
scanf("%d", &n);
```

```
while(n>=41)
{          printf("\nPor motivos de espacio esta juego no permite un tamanho mayor a 40\n");
           printf("\n1-Introduzca el tamanho de la sopa(matriz simetrica): ");
   scanf("%d", &n);    }
printf("\n2-Introduzca el numero de palabras: ");            //cantidad de palabras
scanf("%d", &k);
while(k>n)
{
printf("\nEl numero de palabras debe ser menor o igual al tamanho de la sopa\n2-Introduzca el numero de palabras: ");
scanf("%d", &k);
}
for(i=0;i<k;i++)                                                    //ingresar palabras
{
printf("\n-Escriba la palabra %d: ", i+1);
scanf("%s",palabras);
largo=ncaracteres(palabras);
for(j=0;j<largo;j++)
{
mpalabras[i][j]=palabras[j];
}
repetir=0;
while(repetir==0)
{
ejex[i]=aleatorio(n,largo);                  //coordenadas aleatorias de inicio de la palabra
ejey[i]=aleatorio(n,largo);
x=ejex[i];
y=ejey[i];
direccion=rand()%4;                          //condiciones para definir direccion de la palabra
if(direccion==0)
{
horizontal(sopa,palabras,largo,x,y,&repetir);      //CASO 0
}
if(direccion==1)
{
vertical(sopa,palabras,largo,x,y,&repetir);        //CASO 1
}
if(direccion==2)
{
diagonalp(sopa,palabras,largo,x,y,&repetir);       //CASO 2
}
```

```
if(direccion==3)
{
diagonals(sopa,palabras,largo,x,y,&repetir,n);     //CASO 3
}
}
}
for(p=0;p<n;p++)                          //for matriz auxiliar para guardar celdas ocupadas
{
for(q=0;q<n;q++)
{
if(sopa[p][q]!='\0')
{
casillas[p][q]=1;
}
}
system("cls");                    //borrar pantalla
printf("\n");
}
for(i=0;i<n;i++)                          //for llenar matriz con letras aleatorias
{
for(j=0;j<n;j++)
{
if(casillas[i][j]!=1)                    //condicion para evitar sobrescribir sobre una palabra
{
sopa[i][j]=b+rand()%26;
}
}
}
printf("      ");
for(p=0;p<n;p++)
{
if(p<=9)
{
printf("[%d] ",p);
}
else
{
printf("[%d]",p);
}
}
```

```
printf("\n\n");
for(p=0;p<n;p++)                        //for imprimir matriz sopa
{
if(p<=9)
{
printf(" [%d] ",p);
}
else
{
printf("[%d] ",p);
}
for(q=0;q<n;q++)
{
printf(" | %c",sopa[p][q]);
}
printf(" |\n");
}
//AQUI COMIENZA EL JUEGO//
time_t inicio, fin;
inicio=time(NULL);
printf("\n\n*****BUSQUE PALABRAS EN LA SOPA DE LETRAS E INGRESELAS, RECUERDE QUE SOLO PUEDE COMETER 3 ERRORES!*****\n");
i=3;
j=k;
while(i>0)
{
printf("\nIngrese una palabra: ");
scanf("%s",palabras);
largo=ncaracteres(palabras);
respuesta=revisar(mpalabras,palabras,largo,k,&num);
if(respuesta==1)
{  printf("\nLa palabra '%s' SI se encuentra en la sopa y comienza en la coordenada (%d,%d)\n",palabras,ejex[num],ejey[num]);
        j=j-1;
        if(j==0)
        {   i=0;
                printf("\nFELICIDADES HAS ADIVINADO TODAS LAS PALABRAS!!!");
                fin=time(NULL);         //fin del programa
                printf("\nTIEMPO UTILIZADO: %f", difftime(fin,inicio));  //tiempo utilizado
        }
}
```

```
if(respuesta==0)
{
	printf("\nLa palabra '%s' NO se encuentra en la sopa\n",palabras);
	i=i-1;
	if(i==0)
	{
		printf("\nCOMETISTE 3 ERRORES, GAME OVER");
		fin=time(NULL);
		printf("\nTIEMPO UTILIZADO: %f", difftime(fin,inicio));
	}
}
}
} //*****************************FIN DEL MAIN*****************************//
//*****************************FUNCIONES*********************************//
int revisar(char mpalabras[100][100], char palabras[30], int largo, int k,int *num)
{
	int i,j,a,count;		//verifica que las palabras se encuentren en la sopa
	int respuesta=0;

	for(i=0;i<k;i++)
	{
		a=0;
		count=largo;
		for(j=0;j<largo;j++)
		{
			if(mpalabras[i][j]==palabras[a])
			{
			count=count-1;
			a=a+1;
			if(count==0)
			{
				respuesta=1;
				*num=i;
			}
			}
		}
	}
	return(respuesta);
}
```

```
int aleatorio(int n, int largo)             //busca numeros aleatorios entre 0 y n-largo
{
int random;
random=rand()%(n-largo);
return(random);
}
int ncaracteres(char palabra[30])           //contador de caracteres
{
int count;
count=0;
while(palabra[count] != '\0')
            {
            count=count+1;
            }
return(count);
}
void horizontal(char sopa[100][100],char palabras[30],int largo,int x, int y,int *repetir)
{                   //CASO 0,  ingresa palabra de forma horizontal en la sopa
int a;

            for(a=0;a<largo;a++)
{
   if(sopa[x][y]==palabras[a] || sopa[x][y]=='\0')
   {
   sopa[x][y]=palabras[a];
            y=y+1;
            if(a==largo-1)
            {
            *repetir=1;
            }
            }
            else
            {
            *repetir=0;
            }
}
}
```

```
void vertical(char sopa[100][100],char palabras[30],int largo,int x, int y,int *repetir)
{       //CASO 1,  ingresa palabra de forma vertical en la sopa
int a;
for(a=0;a<largo;a++){
   if(sopa[x][y]==palabras[a] || sopa[x][y]=='\0') {
   sopa[x][y]=palabras[a];
            x=x+1;
            if(a==largo-1){
            *repetir=1;    }
            }
            else
            {
            *repetir=0;
            }
}
}
void diagonalp(char sopa[100][100],char palabras[30],int largo,int x, int y,int *repetir)
{                   //CASO 2,  ingresa palabra de forma en la diagonal principal de la sopa
int a;
            for(a=0;a<largo;a++)
{
   if(x!=y)
   {
   break;
            }
   if(sopa[x][y]==palabras[a] || sopa[x][y]=='\0')
   {
   sopa[x][y]=palabras[a];
            x=x+1;
            y=y-1;
            if(a==largo-1)
            {
            *repetir=1;
            }
            }
            else
            {
            *repetir=0;
            }
}
}
```

```
void diagonals(char sopa[100][100],char palabras[30],int largo,int x, int y,int *repetir,int n)
{                       //CASO 2,  ingresa palabra de forma en la diagonal secundaria de la sopa
int a;
        for(a=0;a<largo;a++)
{
   if(x+y!=n-1)
   {
   break;
        }
   if(sopa[x][y]==palabras[a] || sopa[x][y]=='\0')
   {
   sopa[x][y]=palabras[a];
        x=x+1;
        y=y+1;
        if(a==largo-1)
        {
        *repetir=1;
        }
        }
        else
        {
        *repetir=0;
        }
}
}
```

En cuanto a la tarea de la sopa de letras cabe destacar que fue necesario hacer uso de la mayoría de las herramientas aprendidas en la asignatura, ya sean bifurcaciones, ciclos, arreglos, matrices, y funciones. La primera dificultad destacable en un principio fue el introducir las palabras dentro de la matriz de forma correcta en las distintas direcciones que estas podían tomar, limitar las coordenadas para que las palabras no salgan de la matriz y no se intercepten sobrescribiendo otra palabra fue la mayor dificultad lo cual pude buscar una solución para varios casos con la utilización de parámetros por referencia.