

Ejercicios del Capítulo 6

Práctica 6

M. A. Noriega Vargas, C. D. Ruiz Guerrero,
A. Ruiz Medina, A. Rodríguez Buenrostro

1 Objetivo

-Realizar una serie de programas con funciones recursivas.

2 Introducción

La recursividad es una técnica que se utiliza en programación que le permite al programador que un bloque de instrucciones se ejecute un cierto número de veces que este ha indicado. Se dice que un función es recursiva cuando esta se llama así misma. Un algoritmo recursivo es un algoritmo que expresa la solución de un problema en términos de una llamada a sí mismo. La llamada a sí mismo se conoce como llamada recursiva o recurrente.

3 Ejercicios

3.1 Suma de los elementos de un arreglo

```
#include <stdio.h>

int suma(int a[],int n,int i);

int main(){

    int a[1000],i,n,s;

    printf("Ingrese el tamaño del arreglo:");
    scanf("%d", &n);

    printf("Ingrese los elementos del arreglo: ");
    for(i=0; i<n; i++){
```

```

scanf("%d", &a[i]);
    }

s=suma(a,n,0);
printf("La suma del arreglo es :%d \n",s);
return 0;
}

int suma(int a[],int n,int i){
    if(i<n){
        return a[i]+suma(a,n,++i);
    }
    return 0;
}

```

```

mié 18/12
laptop01@laptop01-Aspire-E5-511P: ~/Practicas/Capitulos
Archivo Editar Ver Buscar Terminal Ayuda
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$ gcc suma_arreglo.c
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$ ./a.out
Ingrese el tamaño del arreglo:4
Ingrese los elementos del arreglo: 1 2 3 5
La suma del arreglo es :11
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$

```

3.2 Impresión de un arreglo

```

#include <stdio.h>
#define MAX 100

void Elementos(int a[], int st, int l);

int main(){
    int a[MAX];
    int n, i;

    printf("Ingrese el numero de elementos del arreglo:");
    scanf("%d",&n);

    printf("Ingrese los %d elementos del arreglo:\n",n);

```

```

for(i=0;i<n;i++){
    printf(" elemento %d: ",i);
    scanf("%d",&a[i]);
}

printf("Los elementos en el arreglo son: ");
Elementos(a, 0, n);

return 0;
}

void Elementos(int a[], int st, int l){
    if(st >= l){
        return;
    }
    printf("%d ", a[st]);
    Elementos(a, st+1, l);
}

```

```

mié 18:13
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$ gcc imprintr_Arreglo.c
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$ ./a.out
Ingrese el numero de elementos del arreglo:7
Ingrese los 7 elementos del arreglo:
elemento 0: 1
elemento 1: 4
elemento 2: 7
elemento 3: 8
elemento 4: 5
elemento 5: 2
elemento 6: 9
Los elementos en el arreglo son: 9 2 5 8 7 4 1
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$

```

3.3 Impresión inversa de un arreglo

```

#include<stdio.h>

void invertir(int a[], int i, int f);
void Elementos(int a[], int st, int l);

int main(){
    int a[100], i, n;

    printf("Ingrese el numero de elementos del arreglo:");

```

```

scanf("%d",&n);

printf("Ingrese los %d elementos del arreglo:\n",n);
for(i=0;i<n;i++){
    printf(" elemento %d: ",i);
    scanf("%d",&a[i]);
}

invertir(a, 0, n-1);
printf("El arreglo invertido es: \n");
Elementos(a,0, n);
return 0;
}

void invertir(int a[], int i, int f){
    int c;
    if (i < f){
        c = a[i];
        a[i] = a[f];
        a[f] = c;
        invertir(a, ++i, --f);
    }
}

void Elementos(int a[], int st, int l){
    if(st >= l){
        return;
    }
    printf("%d \n", a[st]);
    Elementos(a, st+1, l);
}

```

```

mié 18:13
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$ gcc imprimir_Arreglo_invertido.c
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$ ./a.out
Ingrese el numero de elementos del arreglo:6
Ingrese los 6 elementos del arreglo:
elemento 0: 3
elemento 1: 5
elemento 2: 7
elemento 3: 1
elemento 4: 4
elemento 5: 8
El arreglo invertido es:
8
4
1
7
5
3
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$

```

3.4 Impresión inversa de una cadena

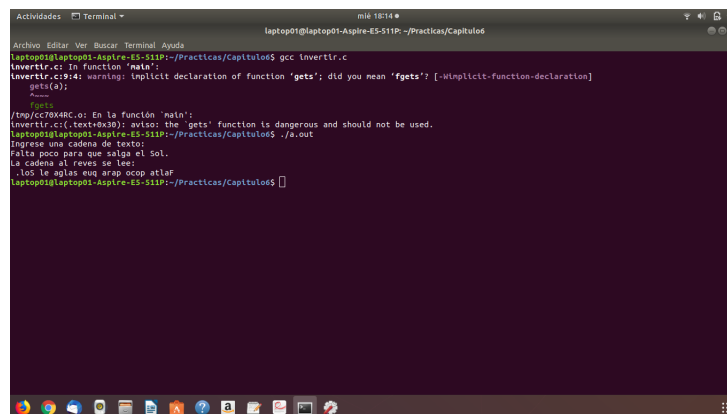
```
#include <stdio.h>
#include <string.h>

void invertir(char *x, int i, int f);

int main(){
    char a[100];
    printf("Ingrese una cadena de texto:\n");
    gets(a);
    invertir(a, 0, strlen(a)-1);
    printf("La cadena al revés se lee:\n %s\n", a);
    return 0;
}

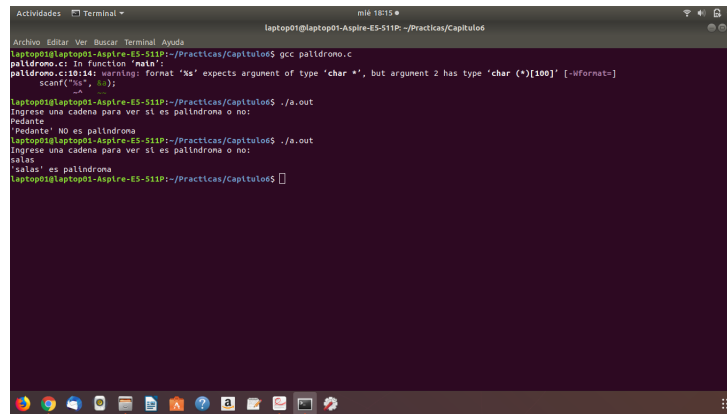
void invertir(char *x, int i, int f){
    char c;

    if (i >= f)
        return;
    c = *(x+i);
    *(x+i) = *(x+f);
    *(x+f) = c;
    invertir(x, ++i, --f);
}
```



```
mié 18:14
laptop01@laptop01-Aspire-E5-511P: ~/Practicas/Capitulos
Invertir.c: In function 'main':
Invertir.c:14: warning: implicit declaration of function 'gets'; did you mean 'fgets'? [-Wimplicit-function-declaration]
    gets(a);
    ^~~~~
/tmp/cc7BK4BC.o: En la función 'main':
Invertir.c:(.text+0x30): aviso: the 'gets' function is dangerous and should not be used.
laptop01@laptop01-Aspire-E5-511P: ~/Practicas/Capitulos$ ./a.out
Ingrese una cadena de texto:
La cadena al revés se lee:
lo5 le aglas euq arap ocap atlar
laptop01@laptop01-Aspire-E5-511P: ~/Practicas/Capitulos$
```

3.5 Verificar si una cadena es un palíndromo



The screenshot shows a terminal window with the following commands and output:

```
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$ gcc palindromo.c
palindromo.c: In function 'main':
palindromo.c:10:14: warning: format '%s' expects argument of type 'char *', but argument 2 has type 'char (*)[100]' [-Wformat=]
    scanf("%s", i);
               ^
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$ ./a.out
Ingrese una cadena para ver si es palindroma o no:
Pedante
'Pedante' NO es palindroma
Ingrese una cadena para ver si es palindroma o no:
salas
'salas' es palindroma
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$
```

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

int Palindromo(char *a, int i, int f);

int main() {
    char a[100];
    int n,r;
    printf("Ingrese una palabra para ver si es palindroma o no:\n");
    scanf("%s", &a);

    n = strlen(a);
    r = Palindromo(a, 0, n-1);

    if (r) {
        printf("%s' es palindroma\n", a);
    }
    else{
        printf("%s' NO es palindroma\n", a);
    }
}

int Palindromo(char *a, int i, int f){
    if (i>= f){
        return 1;
    }

    if (a[i] == a[f]) {
        return Palindromo(a, ++i, --f);
    }
}
```

```

    }
else {
    return 0;
}
}

```

3.6 Valor mínimo de un arreglo

```

#include <stdio.h>
#define MAX_SIZE 100

int Min(int array[], int index, int len);

int main(){
    int array[MAX_SIZE], n, max, min;
    int i;

    // Inputting size and elements of array
    printf("Ingrese el numero de elementos del arreglo: ");
    scanf("%d", &n);
    printf("Ingrese los %d elementos del arreglo:\n",n);
    for(i=0; i<n; i++)
    {
        printf(" elemento %d : ",i);
        scanf("%d", &array[i]);
    }

    min = Min(array, 0, n);

    printf("EL valor minimo en el arreglo es: %d\n", min);

    return 0;
}

int Min(int array[], int index, int len)
{
    int min;

    if(index >= len-2)
    {
        if(array[index] < array[index + 1])
            return array[index];
        else
            return array[index + 1];
    }
}

```

```

min = Min(array, index + 1, len);

if(array[index] < min)
    return array[index];
else
    return min;
}

```

```

Actividades Terminal * mié 18/10
laptop01@laptop01-Aspire-E5-511P ~/Practicas/Capitulos
Archivo Editar Ver Buscar Terminal Ayuda
laptop01@laptop01-Aspire-E5-511P ~/Practicas/Capitulos$ gcc ValMin.c
laptop01@laptop01-Aspire-E5-511P ~/Practicas/Capitulos$ ./a.out
Ingrese el numero de elementos del arreglo: 11
Ingrese los 11 elementos del arreglo:
elemento 0 : -2
elemento 1 : 1
elemento 2 : 0
elemento 3 : 8
elemento 4 : 100
elemento 5 : -9
elemento 6 : 2
elemento 7 : 98
elemento 8 : 601
elemento 9 : 85
elemento 10 : 2222
El valor minimo en el arreglo es: -9
laptop01@laptop01-Aspire-E5-511P ~/Practicas/Capitulos$

```

3.7 Búsqueda binaria

```

#include <stdio.h>

int binarySearch(int*, int, int, int, int);

int main()
{
    int arr1[10], i, n, md, c, low, hg;

    printf("Ingrese el numero de elementos del arreglo:");
    scanf("%d", &n);
    printf("Ingrese los %d elementos del arreglo:\n", n);
    for (i = 0; i < n; i++)
    {
        printf(" element - %d : ", i);
        scanf("%d", &arr1[i]);
    }
    printf("Ingrese el numero que desea buscar: ");
    scanf("%d", &md);
    low = 0, hg = n - 1;

```



```

        c = binarySearch(arr1, n, md, low, hg);
        if (c == 0)
            printf(" El numero no existe en el arreglo.\n\n");
        else
            printf(" Se encontro el numero en el arreglo.\n\n");
        return 0;
    }

int binarySearch(int arr1[], int n, int md, int low, int hg)
{
    int mid, c = 0;
    if (low <= hg)
    {
        mid = (low + hg) / 2;
        if (md == arr1[mid])
        {
            c = 1;
        }
        else if (md < arr1[mid])
        {
            return binarySearch(arr1, n, md, low, mid - 1);
        }
        else
            return binarySearch(arr1, n, md, mid + 1, hg);
    }
    else
        return c;
}

```

```

mié 18:18
laptop01@laptop01-Aspire-E5-511P: ~/Practicas/Capitulos
Archivo Editar Ver Buscar Terminal Ayuda
laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$ ./a.out
Ingrese el numero de elementos del arreglo:10
Ingrese los 10 elementos del arreglo:
element - 0 : 9
element - 1 : 8
element - 2 : 7
element - 3 : 6
element - 4 : 5
element - 5 : 4
element - 6 : 3
element - 7 : 2
element - 8 : 1
element - 9 : 0
Ingrese el numero que desea buscar: 5
Se encontro el numero en el arreglo.

laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$ ./a.out
Ingrese el numero de elementos del arreglo:8
Ingrese los 8 elementos del arreglo:
element - 0 : 80
element - 1 : 81
element - 2 : 82
element - 3 : 83
element - 4 : 84
element - 5 : 85
element - 6 : 86
element - 7 : 87
Ingrese el numero que desea buscar: 8
El numero no existe en el arreglo.

laptop01@laptop01-Aspire-E5-511P:~/Practicas/Capitulos$

```