

**UNIVERSIDAD SERGIO ARBOLEDA**  
**ESCUELA DE CIENCIAS EXACTAS E INGENIERÍA**  
**Ciencias de la computación e Inteligencia Artificial**  
**Arquitectura de Computadores**  
**PARCIAL II**

Estudiantes: Miguel Salazar, Diego Alejandro Bermúdez

**Introducción:**

Este taller-parcial tiene como propósito interiorizar los principios básicos de la programación en el lenguaje C, tales como manejo de variables, estructuras de control, manipulación de strings, manejo de archivos, implementación de algoritmos clásicos, entre otros. El criterio de selección de los ejercicios, por consiguiente, se basó en abarcar la mayor cantidad de elementos, de tal manera que a los autores permitiera sentirse cómodos con la programación en dicho lenguaje de programación.

**Instrucciones de Instalación:**

**1. Clonar el Repositorio parcial2Arquitectura (rama 'master') en un nuevo fichero**

El repositorio está ubicado en el siguiente link:

[Repositorio \(Click aquí\)](#)

En caso de no funcionar, copiar y pegar el siguiente enlace en el navegador:

<https://github.com/miguelsalazar88/parcial2Arquitectura>

2. En una pestaña de Terminal con ubicación en el fichero, ejecutar el Makefile mediante el comando *make*.
3. Elegir cualquiera de los ejecutables (por ejemplo sumaDeArray) y ejecutarlo en Terminal mediante el comando `./<nombre del ejecutable>` . Por ejemplo: `./sumaDeArray`

**Ejercicios Implementados:**

A continuación una breve explicación de cada ejercicio implementado, junto con su debida captura de pantalla de la ejecución:

## Sección 1: “Simple C Questions”

### 1.1 Simple Interest

Ejercicio:	“Simple Interest”
Nombre Archivo:	<b>interesSimple.c</b>
Descripción:	<p>Este programa calcula el interés simple de un préstamo. Se le pide al usuario el valor del préstamo, la tasa de interés (%) y el tiempo con base en la fórmula:</p> $\text{Valor total} = \text{Valor inicial} + \frac{(\text{Valor préstamo}) * (\text{Tasa de Interés}) * (\text{Tiempo})}{100}$
¿Requiere entrada de datos del usuario?	Sí.
Función Principal:	double interesSimple
Entrada:	unsigned long valor, int tasa, int tiempo
Salida:	double valorInteres

Captura de pantalla:

```
**** Bienvenido al programa que calcula el interés simple de un préstamo ****

Por favor ingrese el valor del préstamo en pesos: $100000
Por favor ingrese la tasa de interés en porcentaje: 12
Por favor ingrese el tiempo en años: 4
-----
El valor del interés a pagar será de: $ 48000.000000
El gran total a pagar será de: $ 148000.000000
Miguels-MacBook-Pro:parcial2Arquitectura miguelsalazar$ █
```

## 1.2 Factorial of a Given Number

Ejercicio:	"Factorial of a Given Number"
Nombre Archivo:	<b>factorial.c</b>
Descripción:	Este es un programa que calcula el factorial de un número entero positivo ingresado por el usuario, mediante la utilización de una estructura de control for.
¿Requiere entrada de datos del usuario?	Sí.
Función Principal:	int factorial(int numero)
Entrada:	int numero
Salida:	int resultado

### Captura de Pantalla:

```
**** Bienvenido al programa para calcular el factorial de un número entero positivo ****  
Escriba un número entero positivo para calcular su factorial:5  
El factorial de 5 es: 120
```

### 1.3 Area of a Circle

Ejercicio:	"Area of a Circle"
Nombre Archivo:	<b>areaCirculo.c</b>
Descripción:	Este es un programa que calcula el área de un círculo partiendo de su radio.
¿Requiere entrada de datos del usuario?	Sí.
Función Principal:	float area(float radio)
Entrada:	float radio
Salida:	float area

#### Captura de Pantalla:

```
**** Bienvenido al programa para calcular el area de un circulo ****  
  
Escriba un número de coma flotante que haga referencia a el radio del circulo para calcular su area:3.14  
El area de el circulo con radio 3.140000 es: 30.974844  
  
-----  
Process exited after 13.55 seconds with return value 0  
Press any key to continue . . .
```

## Sección 2: “If/Else Statement”

### 2.1 The Number is Even or Odd

Ejercicio:	“The Number is Even or Odd”
Nombre Archivo:	<b>parOImpar.c</b>
Descripción:	Este programa verifica si un número entero ingresado por el usuario es par o impar. Toma como base el funcionamiento del operador aritmético del residuo (%), dado que el residuo de un número par dividido dos es siempre igual a cero.
¿Requiere entrada de datos del usuario?	Sí.
Función Principal:	esPar(int num)
Entrada:	int num
Salida:	bool esPar

#### Captura de Pantalla:

```
---- Bienvenido al programa que calcula si un número es par o impar ----  
Introduzca un número entero: 153  
El número 153 es impar.
```

## 2.2 Find The Maximum Between Two Numbers

Ejercicio:	"Find The Maximum Between Two Numbers"
Nombre Archivo:	<b>numeroMayor.c</b>
Descripción:	Este programa verifica cuál número (de dos ingresados por el usuario) es mayor.
¿Requiere entrada de datos del usuario?	si
Función Principal:	void max(int a, int b)
Entrada:	int a, int b
Salida:	void (La función no retorna ningún valor pero se muestra el número que sea mayor).

### Captura de Pantalla:

```
**** Bienvenido al programa para el numero mayor entre dos numeros enteros****
Escriba el primer numero:45
Escriba el segundo numero:-4
El numero mayor entre 45 y -4 es:
45
-----
Process exited after 5.845 seconds with return value 0
Press any key to continue . . .
```

## 2.3 Input Any Alphabet and Check Whether it is Vowel or Consonant

Ejercicio:	"Input any Alphabet and Check Whether it is Vowel or Consonant"
Nombre Archivo:	<b>vocalOconsonante.c</b>
Descripción:	Este programa verifica si un caracter ingresado por el usuario es vocal o consonante. Se utiliza la estructura de control if que verifica si el caracter corresponde a ('a'    'e'    'i'    'o'    'u').
¿Requiere entrada de datos del usuario?	Sí.
Función Principal:	void vocalOConsonante(char c)
Entrada:	char c
Salida:	void

### Captura de Pantalla:

```
**** Bienvenido al programa que verifica si un caracter es vocal o consonante ****
Por favor ingrese un caracter en lower case: j
El caracter ingrasado es una consonante.
```

## Sección 3: “Loops”

### 3.1 Reverse a Given Number

Ejercicio:	“Reverse a Given Number”
Nombre Archivo:	<b>reversoDeUnNumero.c</b>
Descripción:	Este programa muestra el reverso de un número ingresado por el usuario.
¿Requiere entrada de datos del usuario?	si
Función Principal:	int rev(int numero)
Entrada:	int numero
Salida:	int reverso

Captura de Pantalla:

```
**** Bienvenido al programa que muestra el reverso de un numero ****  
  
Por favor introduzca el número entero positivo al que desea ver su reverso: 1234  
El reverso del numero: 1234 es: 4321  
-----  
Process exited after 3.047 seconds with return value 0  
Press any key to continue . . .
```



### 3.2 Display Fibonacci Series

Ejercicio:	"Display Fibonacci Series"
Nombre Archivo:	<b>fibonacci.c</b>
Descripción:	<p>Este programa muestra la cantidad de términos de la serie de Fibonacci introducida por el usuario. Utiliza la estructura de control while de la siguiente manera, expuesta en pseudocódigo:</p> <pre>num1 = 0 num2 = 1 i  While ( i &lt; inputUsuario) do     suma = num1+num2     num2 = suma     num1 = num2 end</pre>
¿Requiere entrada de datos del usuario?	Sí.
Función Principal:	No se utilizan funciones en este ejercicio
Entrada:	int n
Salida:	int suma

Captura de Pantalla:

```
**** Bienvenido al programa que muestra la serie de Ficobacci ****
Por favor introduzca el número de términos de la serie que desea ver: 10
0  1  1  2  3  5  8  13  21  34
Miguels-MacBook-Pro:parcial2Arquitectura miguelasalazar$
```

### 3.3 Print Multiplication Table Using (For Loop)

Ejercicio:	"Print Multiplication Table Using (For Loop)"
Nombre Archivo:	<b>tablaMultiplicacion.c</b>
Descripción:	Este programa, mediante la implementación de una estructura de control tipo for, itera entre los números 1 y 10, y los multiplica por el número ingresado por el usuario para de esta manera imprimir su tabla de multiplicación.
¿Requiere entrada de datos del usuario?	Sí.
Función Principal:	void tablaMultiplicacion(int n)
Entrada:	int n
Salida:	Void (imprime $n \times i$ en cada iteración)

Captura de Pantalla:

```
**** Bienvenido al programa que imprime las tablas de multiplicar.

Ingrese por favor el número del que desea ver la tabla de multiplicar: 14
14 X 1 = 14
14 X 2 = 28
14 X 3 = 42
14 X 4 = 56
14 X 5 = 70
14 X 6 = 84
14 X 7 = 98
14 X 8 = 112
14 X 9 = 126
14 X 10 = 140
```

## Sección 4: Switch Case

### 4.1 Calculator

Ejercicio:	"Calculator"
Nombre Archivo:	<b>calculadora.c</b>
Descripción:	Este programa, mediante la implementación de la estructura condicional Switch, realiza operaciones entre dos números ingresados por el usuario.
¿Requiere entrada de datos del usuario?	Sí.
Funciones:	int suma(int uno, int dos) int resta(int uno, int dos) int multiplicacion(int uno, int dos) double división(int uno, int dos)
Entrada:	int uno, int dos
Salida:	int suma, int resta, int multiplicacion, double division

Captura de Pantalla:

```
**** Bienvenido al programa que hace operaciones con dos números ****  
  
Ingrese el primer número entero: 10  
  
Ingrese el segundo número entero: 5  
Elija la operación que desea hacer con los dos números:  
1. Suma  
2. Resta  
3. Multiplicación  
4. División  
3  
La multiplicación entre 10 y 5 es igual a 50
```

## 4.2 Remove All Vowels From a String

Ejercicio:	"Remove all Vowels Form a String"
Nombre Archivo:	<b>removerVocales.c</b>
Descripción:	Este programa, mediante la implementación de una estructura condicional Switch, reemplaza todos aquellos caracteres de un string (introducido por el usuario) que correspondan a una vocal por un caracter vacío ( ' \0' ). Posteriormente, se imprime el arreglo omitiendo los caracteres vacíos.
¿Requiere entrada de datos del usuario?	Sí.
Función Principal:	void removerVocales(char string[ ])
Entrada:	char string[ ]
Salida:	void

Captura de Pantalla:

```
**** Bienvenido al programa que remueve todas las vocales de una palabra ****  
Por favor ingrese la palabra: Profesor  
Palabra original: Profesor  
Palabra sin vocales: Prfsr
```

### 4.3 Find The Maximum Between Two Numbers Using Switch Case

Ejercicio:	"Find the Maximum Between Two Numbers Using Switch Case"
Nombre Archivo:	<b>numeroMayorSwitchCase.c</b>
Descripción:	Este programa determina cual es el número mayor (de dos ingresados) empleando Switch Case
¿Requiere entrada de datos del usuario?	si
Funciones:	void max(int a,int b)
Entrada:	int a,b
Salida:	void "muestra que numero es mayor o si son iguales"

#### Captura de Pantalla:

```
**** Bienvenido al programa para el numero mayor entre dos numeros enteros****
Escriba el primer numero:7
Escriba el segundo numero:-9
El numero mayor entre 7 y -9 es:
7
-----
Process exited after 3.956 seconds with return value 0
Press any key to continue . . .
```

## Sección 5: “Array Questions”

### 5.1 Remove Duplicate Items in An Array

Ejercicio:	“Remove Duplicate Items In An Array”
Nombre Archivo:	<b>removerRepetidosArreglo.c</b>
Descripción:	Este programa elimina los números repetidos de un arreglo predeterminado y muestra el arreglo resultante
¿Requiere entrada de datos del usuario?	no
Funciones:	<code>void remover(int numero[],int n[],int tamaño)</code> <code>int contiene(int n,int numeros[],int tamaño)</code>
Entrada:	“Ingresa un arreglo predeterminado”
Salida:	“Sale el arreglo sin números repetidos”

#### Captura de Pantalla:

```
El arreglo
1 36 40 8 2 3 40 30 9 10 15 2
Sin numeros repetidos es:
1 36 40 8 2 3 30 9 10 15
-----
Process exited after 0.07805 seconds with return value 0
Press any key to continue . . .
```

## 5.2 Sum of All Array Elements Using Recursion

Ejercicio:	"Sum of All Array Elements Using Recursion"
Nombre Archivo:	<b>sumaDeArray.c</b>
Descripción:	En este programa se utiliza una función recursiva para sumar todos los elementos de un arreglo cuyos valores son introducidos por el usuario. Toma como caso base que el índice (int inicio) sea igual al tamaño del arreglo. De no ser así, procede a llamar nuevamente a la función.
¿Requiere entrada de datos del usuario?	Sí.
Función Principal:	int sumaDeArray(int arr[ ], int inicio, int tamaño)
Entrada:	int arr[ ], int inicio, int tamaño
Salida:	(arr[inicio] + sumaDeArray(arr, inicio+1, tamaño)) recursivamente hasta llegar al caso base.

### Captura de Pantalla:

```
**** Bienvenido al programa que suma todos los elementos de un arreglo ****

Por favor ingrese el tamaño del arreglo (entre 1 y 10):5
Ingrese el valor del elemento 1 del arreglo: 4
Ingrese el valor del elemento 2 del arreglo: 8
Ingrese el valor del elemento 3 del arreglo: 7
Ingrese el valor del elemento 4 del arreglo: 2
Ingrese el valor del elemento 5 del arreglo: 9

La suma de todos los elementos del arreglo es 30
Miguels-MacBook-Pro:parcial2Arquitectura miguelsalazar$
```

### 5.3 Delete All Duplicate Elements From An Array

Ejercicio:	"Delete All Duplicate Elements From An Array"
Nombre Archivo:	<b>removerDuplicados.c</b>
Descripción:	En este programa se utilizan tres estructuras if anidadas para lograr remover los elementos duplicados: El primero tiene como propósito recorrer el arreglo, el segundo es utilizado para comparar los elementos $i$ y $j$ , y el tercero tiene el propósito desplazar hacia la izquierda todos los elementos que suceden el elemento duplicado.
Función Principal:	void removerDuplicados(int arr[ ], int tamaño)
Entrada:	int arr[ ], int tamaño
Salida:	void

#### Captura de Pantalla:

```
**** Bienvenido al programa que remueve elementos duplicados en un arreglo ****  
  
Arreglo original:  
1 2 3 1 4 2 1 6 1  
Arreglo sin elementos repetidos:  
1 2 3 4 6  
Miguels-MacBook-Pro:parcial2Arquitectura miguelsalazar$
```



## Sección 6: Matrix Questions

### 6.1 Add Two Matrices

Ejercicio:	"Add Two Matrices"
Nombre Archivo:	<b>sumaMatrices.c</b>
Descripción:	Este programa suma dos matrices cuadradas de tamaño ingresado por el usuario generadas por la máquina y muestra su resultado
¿Requiere entrada de datos del usuario?	si
Funciones:	void initMatriz(int n, int matriz[n][n],int y) void imprimirMatriz(int n, int matriz[n][n]) void sumaMatrices(int n, int matriz[n][n],int matriz2[n][n])
Entrada:	int tamano
Salida:	int matriz[tamano][tamano]

### Captura de Pantalla:

```
Por favor digite el tamaño de las matrices: 3
Las matrices son:
1      1      1
2      3      2
3      4      5

3      4      5
4      5      6
5      6      7
La suma de las matrices es igual a:
4      5      6
6      8      8
8      10     12

-----
Process exited after 3.919 seconds with return value 0
Press any key to continue . . .
```

## 6.2 Two Matrices Are Equal or Not

Ejercicio:	"Two Matrices Are Equal or Not"
Nombre Archivo:	<b>matricesIguales.c</b>
Descripción:	En este programa, mediante la utilización de dos estructuras for anidadas, compara tres matrices A, B y C, de las cuales $A = B$ y $A \neq C$ , con el fin de demostrar el correcto funcionamiento del mismo.
¿Requiere entrada de datos del usuario?	No.
Función Principal:	<code>void compararMatrices(int n, int matrizA[n][n], int matrizB[n][n])</code>
Entrada:	<code>int n, int matrizA[n][n], int matrizB[n][n]</code>
Salida:	<code>void</code>

### Captura de Pantalla::

```
Miguels-MacBook-Pro:parcial2Arquitectura miguelsalazar$ gcc matricesIguales.c
Miguels-MacBook-Pro:parcial2Arquitectura miguelsalazar$ ./a.out
**** Bienvenido al programa que compara matrices ****

Se inicializan dos matrices iguales y una diferente a las dos anteriores.
Matriz 1 :
1      1      1
1      1      1
1      1      1

Matriz 2 :
1      1      1
1      1      1
1      1      1

Matriz 3 :
3      3      3
3      3      3
3      3      3

Se comparan las matrices 1 y 2.
Resultado:
Las dos matrices son iguales.

Se comparan las matrices 1 y 3.
Resultado:
Las dos matrices son diferentes.

Miguels-MacBook-Pro:parcial2Arquitectura miguelsalazar$
```

## 6.3 Sum of the Diagonals of A Matrix

Ejercicio:	"Sum of the Diagonals of A Matrix"
Nombre Archivo:	<b>sumaDiagonalMatrices.c</b>
Descripción:	Este programa, mediante la implementación de dos estructuras for, suma los elementos que componen las dos diagonales de una matriz cuadrada. Toma como base el hecho de que todos los elementos de la diagonal 1 cumplen con la regla de que el número de índice de la fila es igual a aquel de la columna ( $i = j$ ) y, asimismo, todos los elementos de la diagonal 2 cumplen con la regla de que la suma de los índices de fila y columna es igual al tamaño de la matriz.
¿Requiere entrada de datos del usuario?	Si.
Función Principal:	int sumaDiagonales(int n,int matriz[n][n]);
Entrada:	int n,int matriz[n][n]
Salida:	int suma

### Captura de Pantalla:

```
Por favor digite el tamaño de la matriz: 5
La matriz es:
1      1      1      1      1
1      1      1      1      1
1      1      1      1      1
1      1      1      1      1
1      1      1      1      1
La suma de las diagonales de la matriz es igual a: 9
Miguels-MacBook-Pro:parcial2Arquitectura miguelasalazar$
```

## Sección 7: “String Questions List”

### 7.1 A String is Palindrome or Not

Ejercicio:	“A String is Palindrome or Not”
Nombre Archivo:	<b>palindroma.c</b>
Descripción:	Este programa lee una palabra, de tamaño elegido, ingresada por el usuario y muestra si es o no palindroma
¿Requiere entrada de datos del usuario?	si
Función Principal:	void palindroma(int tamanio,char palabra[])
Entrada:	int tamanio, char palabra[tamanio]
Salida:	La función no retorna valor pero muestra si la palabra es o no palindroma

#### Captura de Pantalla:

```
Ingrese el tamaño de la palabra que desea analizar si es palindroma o no
7
Ingrese la palabra que desea analizar si es palindroma o no
abafaba
La palabra es palindroma
-----
Process exited after 20.56 seconds with return value 0
Press any key to continue . . .
```

## 7.2 A String is Anagram or Not

Ejercicio:	"A String is Anagram or Not"
Nombre Archivo:	<b>anagrama.c</b>
Descripción:	Este programa captura dos strings de tamaño 6 y muestra si la palabra 1 es anagrama de la palabra 2
¿Requiere entrada de datos del usuario?	si
Funciones:	<code>void anagrama(char palabra1[],char palabra2[])</code> <code>char* contiene(char letra,char palabra[],int tamanio)</code>
Entrada:	<code>char palabra1[6],palabra2[6]</code>
Salida:	Es anagrama o no es anagrama

### Captura de Pantalla:

```
Ingrese la primera palabra (debe tener 6 caracteres)
casala
Ingrese la segunda palabra (debe tener 6 caracteres)
lasaca
La palabra1 es anagrama de la palabra2
-----
Process exited after 11.53 seconds with return value 0
Press any key to continue . . .
```

## 7.3 Compare Two Strings

Ejercicio:	"Compare Two Strings"
Nombre Archivo:	<b>comparaStrings.c</b>
Descripción:	En este programa, mediante la utilización de una estructura de control for, compara índice a índice los elementos que componen los dos strings introducidos por el usuario, y verifica si son iguales o no.
¿Requiere entrada de datos del usuario?	Sí.
Función Principal:	void comparaStrings(int n, char string1[ ], char string2[ ]);
Entrada:	int n, char string1[ ], char string2[ ]
Salida:	bool iguales

### Captura de Pantalla:

```
**** Bienvenido al programa que compara dos palabras ****  
  
Por favor ingrese la primera palabra: Miguel  
Por favor ingrese la segunda palabra: Miguel  
Palabra 1: Miguel  
Palabra 2: Miguel  
Las dos palabras son iguales.  
Miguels-MacBook-Pro:parcial2Arquitectura miguelsalazar$
```

## Sección 8: “String Questions: Level Up”

### 8.1 Find The First Occurrence of a Given Character in a Given String

Ejercicio:	“Find The First Occurrence of a Given Character in a Given String”
Nombre Archivo:	<b>ocurrenciaString.c</b>
Descripción:	En este programa, mediante la utilización de una estructura de control for, se compara el caracter ingresado por el usuario con todos los elementos (índice a índice) que componen el string
¿Requiere entrada de datos del usuario?	Sí
Función Principal:	void primeraOcurrencia(char c, char palabra[ ])
Entrada:	char c, char palabra[ ]
Salida:	void

#### Captura de Pantalla:

```
**** Bienvenido al programa que busca la primera ocurrencia de un caracter en una palabra ****  
  
Por favor ingrese una palabra: anitalavalatina  
Por favor ingrese el caracter que desea buscar: i  
  
La primera ocurrencia de i se encuentra en el indice 2 de la palabra anitalavalatina.  
Miguels-MacBook-Pro:parcial2Arquitectura miguelasalazar$
```

## 8.2 Count Occurrences of a Character in a Given String

Ejercicio:	"Count Occurrences of a Character in a Given String"
Nombre Archivo:	<b>conteoCaracter.c</b>
Descripción:	Este programa cuentan y muestra la cantidad de veces que aparece un caracter ingresado por el usuario en un string de 6 caracteres capturado del mismo modo
¿Requiere entrada de datos del usuario?	si
Función Principal:	int conteo(char palabra[],char letra,int tamaño)
Entrada:	char palabra[6], letra int tamaño
Salida:	int cantidad

### Captura de Pantalla:

```
Ingrese la letra que desea contar
d
Ingrese la palabra (debe tener maximo 6 caracteres)
dadode
La letra d se encuentra 3 veces en la palabra dadode
-----
Process exited after 160.7 seconds with return value 0
Press any key to continue . . .
```



### 8.3 Remove All Repeated Characters From a Given String

Ejercicio:	"Remove All Repeated Characters From a Given String"
Nombre Archivo:	<b>RemoverRepetidos.c</b>
Descripción:	Este programa lee un arreglo de 20 caracteres ingresado por el usuario y lo muestra sin ningún carácter repetido
¿Requiere entrada de datos del usuario?	si
Función Principal:	<code>void remover(char palabra[],char p[],int tamaño)</code> <code>int contiene(char letra,char palabra[],int tamaño)</code>
Entrada:	<code>char palabra[20]</code>
Salida:	<code>char palabra[]</code> (sin caracteres repetidos)

#### Captura de Pantalla:

```
Ingrese la palabra (debe tener 20 caracteres)
juanitavaalatinuuuu
La palabra juanitavaalatinuuuu sin letras repetidas es: juanitvl
-----
Process exited after 27.94 seconds with return value 0
Press any key to continue . . .
```

## Sección 9: “Function Questions”

### 9.1 Find Diameter, Circumference and Area of a Circle Using Functions

Ejercicio:	“Find Diameter, Circumference and Area of a Circle Using Functions”
Nombre Archivo:	<b>funcionesCirculo.c</b>
Descripción:	<p>En este programa, mediante el uso de funciones, se calculan el diámetro, el área y el perímetro de un círculo cuyo radio es ingresado por el usuario. Toma como base las fórmulas básicas de la circunferencia, que son:</p> $d = 2 * r$ $P = 2\pi * r$ $A = \pi * r^2$
¿Requiere entrada de datos del usuario?	Si.
Funciones:	double diametro(double radio) double circunferencia(double radio) double area(double radio)
Entrada:	double radio
Salida:	double diametro, double circunferencia, double area

#### Captura de Pantalla:

```
**** BIENVENIDO AL PROGRAMA QUE CALCULA DIAMETRO, CIRCUNFERENCIA Y AREA DE UN CIRCULO ****
Por favor introduzca el valor del radio del círculo en metros: 23

Resultados:
Diámetro      = 46.000000 metros
Circunferencia = 72.256800 metros
Area          = 1661.906400 metros cuadrados

Miguels-MacBook-Pro:parcial2Arquitectura miguelsalazar$
```

## 9.2 Find The Power of Any Number Using Recursion

Ejercicio:	"Find The Power of Any Number Using Recursion"
Nombre Archivo:	<b>potenciaRecursiva.c</b>
Descripción:	Este programa calcula la potencia entera ingresada de un número entero de manera recursiva
¿Requiere entrada de datos del usuario?	si
Función Principal:	int potenciaR(int numero,int potencia);
Entrada:	int numero,potencia
Salida:	int resultado

### Captura de Pantalla:

```
Ingrese el numero que desea potenciar
5
Ingrese la potencia que desea realizar (int)-3
La potencia -3 del numero 5 es:0.008000
-----
Process exited after 8.627 seconds with return value 0
Press any key to continue . . .
```

### 9.3 Find The Factorial of Any Number Using Recursion

Ejercicio:	"Find The Factorial of Any Number Using Recursion"
Nombre Archivo:	<b>factorialRecursivo.c</b>
Descripción:	Este programa calcula el factorial un número entero positivo de manera recursiva
¿Requiere entrada de datos del usuario?	si
Función Principal:	int factorial(int numero)
Entrada:	int numero
Salida:	int resul

#### Captura de Pantalla:

```
Ingrese el numero al que desea encontrar el factorial (int)
5
El factorial del numero 5 es: 120
-----
Process exited after 2.793 seconds with return value 0
Press any key to continue . . .
```

## Sección 10: “Pointer Questions”

### 10.1 Compare Two Strings using pointers

Ejercicio:	Compare Two Strings using pointers
Nombre Archivo:	<b>comprobarStrings.c</b>
Descripción:	Este programa comprueba que dos strings de máximo 5 caracteres, ingresados por el usuario sean iguales o no, empleando apuntadores.
¿Requiere entrada de datos del usuario?	si
Función Principal:	void verificar(char palabra1[],char palabra2[])
Entrada:	char palabra1[],palabra2[]
Salida:	void “muestra si son o no la misma palabra”

#### Captura de Pantalla:

```
Ingresa la primera palabra (debe tener maximo 5 caracteres)
hola
Ingresa la segunda palabra (debe tener maximo 5 caracteres)
las
El resultado de comprobar la palabra hola con las es:
NO son iguales
-----
Process exited after 5.777 seconds with return value 0
Press any key to continue . . .
```

## 10.2 Concatenate Two Strings using pointers

Ejercicio:	Concatenate Two Strings using pointers
Nombre Archivo:	<b>concatenarApuntadores.c</b>
Descripción:	Este programa concatena dos strings ingresados por el usuario usando apuntadores y muestra su resultado
¿Requiere entrada de datos del usuario?	si
Función Principal:	void concatenar(char palabra1[],char palabra2[])
Entrada:	char palabra1[],palabra2[]
Salida:	void "muestra el resultado de la concatenación"

### Captura de Pantalla:

```
Ingrese la primera palabra (debe tener maximo 5 caracteres)
holass
Ingrese la segunda palabra (debe tener maximo 5 caracteres)
maress
El resultado de concatenar la palabra holass con maress es: holassmaress

-----
Process exited after 8.241 seconds with return value 0
Press any key to continue . . .
```

### 10.3 Find the Length of the String using pointers

Ejercicio:	Find the length of the string using pointers
Nombre Archivo:	<b>tamanoArregloApuntadores.c</b>
Descripción:	Este programa cuenta la cantidad de caracteres ingresados por el usuario en un string, usando apuntadores
¿Requiere entrada de datos del usuario?	si
Función Principal:	int conteo(char palabra1[])
Entrada:	char palabra1[]
Salida:	int i "cantidad de caracteres"

#### Captura de Pantalla:

```
Ingrese la palabra (debe tener maximo 20 caracteres)
jhbgvfc
La palabra jhbgvfc tiene 7 caracteres

-----
Process exited after 5.886 seconds with return value 0
Press any key to continue . . .
```

## Sección 11: File Handling

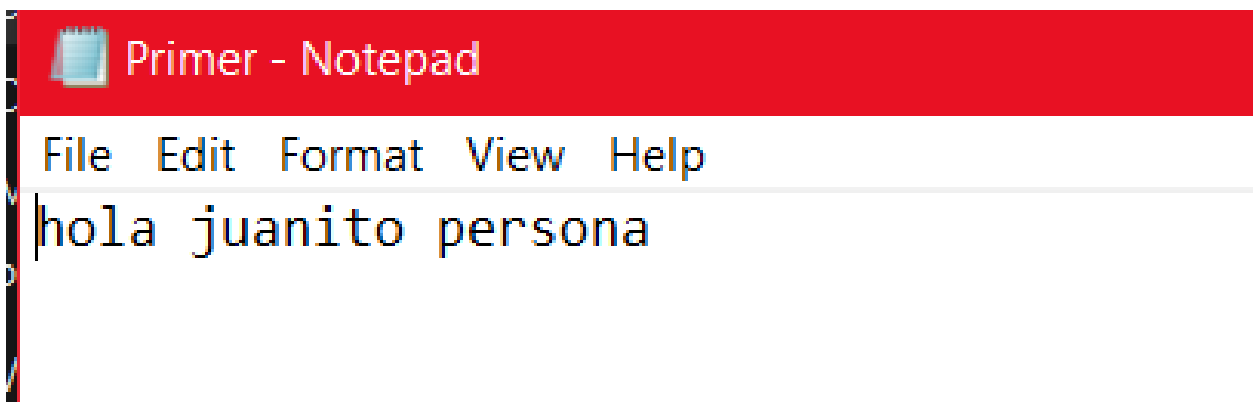
### 11.1 Create a File and Write contents, Save and close the file

Ejercicio:	Create a file and write contents, save and close the file
Nombre Archivo:	<b>manejoArchivos.c</b>
Descripción:	Este programa crea un archivo txt con nombre: Primer escribe en él lo cierra y guarda
¿Requiere entrada de datos del usuario?	si
Función Principal:	main()
Entrada:	char data[] "información a escribir"
Salida:	"Fue guardado correctamente" -- "no se creó el doc"

#### Captura de Pantalla:

```
Ingresa lo que desea guardar en el archivo :
hola juanito persona
Archivo creado y guardado correctamente. :)

-----
Process exited after 13.37 seconds with return value 0
Press any key to continue . . .
```





## 11.2 Read File Contents and Display Them On The Console

Ejercicio:	"Read File Contents and Display Them On The Console"
Nombre Archivo:	<b>leerTexto.c</b>
Descripción:	Este programa, mediante la utilización de un puntero (FILE *), se lee un archivo de texto línea por línea. Cada línea se carga a un arreglo de caracteres que se imprime mediante la utilización de una estructura de control tipo While.
¿Requiere entrada de datos del usuario?	No.
Entrada:	archivo.txt
Salida:	char[ ]

### Capturas de Pantalla:

1. Archivo de texto .t

```
≡ archivoDeTexto.txt
1  Inicio.
2  Este es un archivo de texto.
3  Me van a utilizar para la sección 11 del parcial.
4  Mis autores son Miguel Salazar y Diego Bermúdez.
5  Fin.
```

2. Ejecución del programa:

```

**** Bienvenido al programa que lee archivos de texto. ****

El texto escrito en el archivo archivoDeTexto.txt es:
Inicio.

Este es un archivo de texto.

Me van a utilizar para la sección 11 del parcial.

Mis autores son Miguel Salazar y Diego Bermúdez.

Fin.
Miguels-MacBook-Pro:parcial2Arquitectura miguelsalazar$ █

```

## 11.3 How to Copy Contents From one File to Another

Ejercicio:	“How To Copy Contents From One File to Another”
Nombre Archivo:	<b>copiarContenido.c</b>
Descripción:	En este programa se utilizan dos punteros: uno para la lectura del archivo de texto, y otro para la escritura del mismo texto en otro archivo. Se carga el texto del archivo 1 a un arreglo de caracteres y ese mismo arreglo se utiliza para copiarlo en el archivo 2.
¿Requiere entrada de datos del usuario?	No.
Entrada:	Archivo .txt a char[ ]
Salida:	char[ ] a Archivo .txt

### Captura de Pantalla:

1. Ejecución del programa:

```

**** Bienvenido al programa que lee un archivo de texto y lo escribe en otro archivo vacío ****

El texto escrito en el archivo archivoDeTexto.txt es:
Inicio.

Este es un archivo de texto.

Me van a utilizar para la sección 11 del parcial.

Mis autores son Miguel Salazar y Diego Bermúdez.

Fin.
Revisa ahora el archivo con nombre textoVacio.txt que se encuentra en este fichero.
Miguels-MacBook-Pro:parcial2Arquitectura miguelsalazar$ █

```

2. Resultado en el archivo textoVacio.txt:

```
≡ textoVacio.txt
1  Inicio.
2  Este es un archivo de texto.
3  Me van a utilizar para la sección 11 del parcial.
4  Mis autores son Miguel Salazar y Diego Bermúdez.
5  Fin.
```

## Sección 12: Sorting

### 12.1 Bubble Sort in C

Ejercicio:	"Bubble Sort in C"
Nombre Archivo:	<b>bubbleSort.c</b>
Descripción:	En este programa se implementa el algoritmo clásico denominado "Bubble Sort", con dos de sus mejoras implementadas. Su funcionamiento básico consta en revisar cada elemento de la lista con su vecino derecho e intercambiarlos en el caso de que aquel de la izquierda sea mayor que el de la derecha.
¿Requiere entrada de datos del usuario?	No.
Función Principal:	void bubbleSort(int arreglo[ ], int tamaño)
Entrada:	int arreglo[ ], int tamaño
Salida:	void

### Captura de Pantalla:

Arreglo sin ordenar:

```
a[0] = 1  
a[1] = 7  
a[2] = 6  
a[3] = 8  
a[4] = 9  
a[5] = 0  
a[6] = 4  
a[7] = 2  
a[8] = 5  
a[9] = 3
```

Arreglo ordenado:

```
a[0] = 0  
a[1] = 1  
a[2] = 2  
a[3] = 3  
a[4] = 4  
a[5] = 5  
a[6] = 6  
a[7] = 7  
a[8] = 8  
a[9] = 9
```

## 12.2 Insertion Sort In C

Ejercicio:	"Insertion Sort in C"
Nombre Archivo:	<b>insertionSort.c</b>
Descripción:	Este programa ordena un arreglo de enteros predeterminado mediante el método de inserción
¿Requiere entrada de datos del usuario?	no
Función Principal:	void sortInsertion(int numeros[],int tamaño)
Entrada:	int numeros[]
Salida:	void "muestra el arreglo ordenado"

### Captura de Pantalla:

```

El arreglo
1 36 40 8 2 3 40 30 9 10 15 2
Ordenado es:
1 2 2 3 8 9 10 15 30 36 40 40

-----
Process exited after 0.06667 seconds with return value 0
Press any key to continue . . .

```

### 12.3 Selection Sort in C

Ejercicio:	"Selection Sort in C"
Nombre Archivo:	<b>selectionSort.c</b>
Descripción:	Este programa ordena un arreglo de enteros predeterminado mediante el método de seleccion
¿Requiere entrada de datos del usuario?	no
Función Principal:	void sortSeleccion(int numeros[],int tamaño)
Entrada:	int numeros
Salida:	void "muestra el arreglo ordenado"

#### Captura de Pantalla:

```

El arreglo
1 36 40 8 2 3 40 30 9 10 15 2
Ordenado es:
1 2 2 3 8 9 10 15 30 36 40 40

-----
Process exited after 0.05117 seconds with return value 0
Press any key to continue . . .

```

## Sección 13: Searching

### 13.1 Binary Search

Ejercicio:	Binary Search
Nombre Archivo:	<b>binarySearch.c</b>
Descripción:	En este programa se implementa el algoritmo clásico de búsqueda binaria, que implementa una estrategia conocida como “divide y vencerás”, mediante la cual se divide la estructura principal en la mitad, y subsecuentemente en la mitad de la mitad y así sucesivamente. Solamente se puede utilizar en estructuras que ya se encuentran ordenadas.
¿Requiere entrada de datos del usuario?	Sí.
Función Principal:	void busquedaBinaria(int array[ ], int indexInicio, int indexFinal, int num)
Entrada:	int array[ ], int indexInicio, int indexFinal, int num

Salida:	int
---------	-----

## Captura de Pantalla:

```
**** Bienvenido al programa que implementa el algoritmo de búsqueda binaria.
Por favor ingrese un número entre 1 y 200: 134
1 iteraciones.
2 iteraciones.
3 iteraciones.
4 iteraciones.
5 iteraciones.
6 iteraciones.
7 iteraciones.
Número 134 encontrado después de 8 iteraciones.
Miguels-MacBook-Pro:parcial2Arquitectura miguelasalazar$
```

## 13.2 Linear Search

Ejercicio:	Linear Search
Nombre Archivo:	<b>linearSearch.c</b>
Descripción:	En este programa, por medio de la utilización de una estructura de control for, se efectúa una búsqueda lineal del número 65, que se encuentra ubicado en algún índice del arreglo.
¿Requiere entrada de datos del usuario?	No.
Función Principal:	void busquedaLineal(int n, int tamaño, int arreglo[n]);
Entrada:	int n, int tamaño, int arreglo[ ]
Salida:	void

## Captura de Pantalla:

```
**** Bienvenido al programa que implementa un algoritmo de búsqueda lineal ****
En algún lugar del arreglo se encuentra el número 65, y se buscará mediante este algoritmo.
El arreglo es:
15 29 32 45 65 38 51 90 23 64 86 48 92 46 89 44 33 55 76 10 1 3 5 7 9
Se implementa el algoritmo...
el numero 65 se encuentra en la posición 4
```

## 13.3 Recursive Binary Search in C

Ejercicio:	Recursive Binary Search in C
Nombre Archivo:	<b>busquedaBinariaRecursiva.c</b>
Descripción:	Este programa determina la existencia o ausencia de un numero entero ingresado por el usuario en un arreglo de enteros predeterminado, mostrando la posición en la cual se encuentra el valor
¿Requiere entrada de datos del usuario?	Si.
Función Principal:	void busqueda(int numeros[],int indI,int indF,int x)
Entrada:	int n, int tamaño, int arreglo[ ]
Salida:	int posicion



## Captura de Pantalla:

```
$ Ingrese el numero a buscar
11
Elemento presente en el arreglo en el indice: 7
-----
Process exited after 1.382 seconds with return value 0
Press any key to continue . . .
```