



**Universidad Nacional Autónoma de
México**



Facultad de Ingeniería

Ciudad Universitaria

Estructura de Datos y Algoritmos I

**Actividad 1: Repaso de lo que aprendí en la
asignatura de Fundamentos de Programación.**

López Cruz Marino grupo 15

Conocimientos teóricos acerca de la programación estructurada obtenidos en el semestre 2021-1 en la materia de Fundamentos de programación.

Dentro del temario contenido en la materia que precede a esta, el alumno obtuvo el conocimiento, así como también desarrollo habilidades para diseñar y dar mantenimiento a algoritmos básicos que cumplieran algún tipo de objetivo.

Obtuvo también el conocimiento para definir dichos algoritmos como: una secuencia de instrucciones dadas que representan un modelo de solución para determinado tipo de problema.

También es capaz de describir algunas características de los algoritmos.

- Debe ser preciso, sin lugar a ambigüedades.
- Definido, si se sigue dos o mas veces se debe de llegar al mismo resultado.
- Finito, en algún momento debe de terminar.

Como parte de su conocimiento en algoritmos el alumno posee el conocimiento para etapas básicas de desarrollo de algoritmo.

1. Análisis del problema definición y delimitación.
2. Diseño y desarrollo del algoritmo con ayuda de pseudocódigos y diagramas de flujo.
3. Prueba de algoritmo

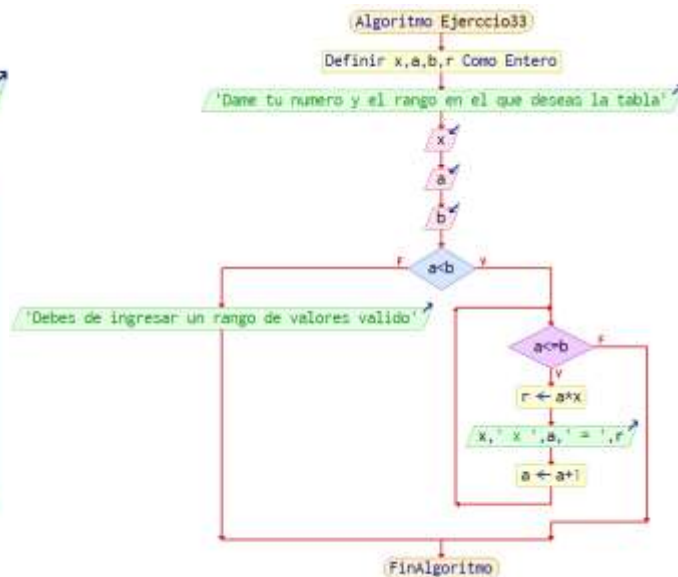
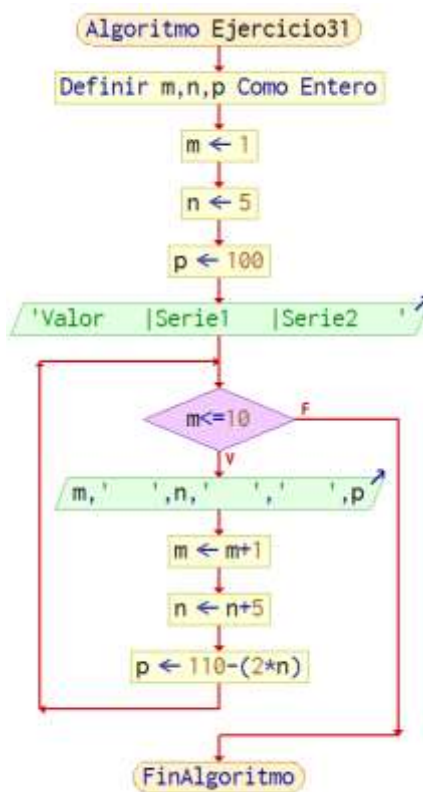
Y como parte del conocimiento obtenido posee habilidades para codificar y llevar los algoritmos a un programa computacional. Siguiendo los siguientes pasos.

1. Codificación
2. Compilación

3. Ejecución
4. Depuración

Diagramas de flujo.

Siendo uno de los temas que abren el temario, es un pilar el conocimiento el desarrollo de los diagramas de flujo. En este caso Pseint es la herramienta de la que se hace uso para facilitar la generación de dichos diagramas, como se muestra en los siguientes ejemplos



Esta es una herramienta que el alumno reconoce como útil a la hora del desarrollo de algoritmos. También conoce claves para facilitar su desarrollo o análisis.

Pseudo códigos.

Así como los diagramas de flujo, los pseudo códigos forman una parte importante de los conocimientos adquiridos en el curso anterior, siendo de una forma similar a como lo hizo con los diagramas de flujo, haciendo uso de la herramienta digital Psint.

Definiendo este como un algoritmo escrito en forma estructurada, que hace uso de palabras en el lenguaje común, utilizando palabras imperativas.

A continuación, se muestra un ejemplo sobre pseudo códigos desarrollados a lo largo del curso.



Códigos y programación en lenguaje C.

Siendo la mitad del curso, la parte de programación codificada en lenguaje C una vez conociendo las bases para la construcción de los algoritmos, se adquiere el conocimiento para la codificación de dichos algoritmos en un lenguaje de alto nivel.

Identifica los tipos de variables y sus propiedades, así como palabras reservadas.

Toda variable en C se declara de la forma: <tipo de dato> <nombre de variable> [, nombre de variable];

En C existen cinco tipos de datos según puede verse en la tabla siguiente:

1. char Carácter o entero pequeño (byte)
2. int Entero
3. float Punto flotante
4. double Punto flotante (mayor rango que float)
5. void Sin tipo (uso especial)

Se conoce la forma en la que funcionan estructuras de repetición o de tipo función

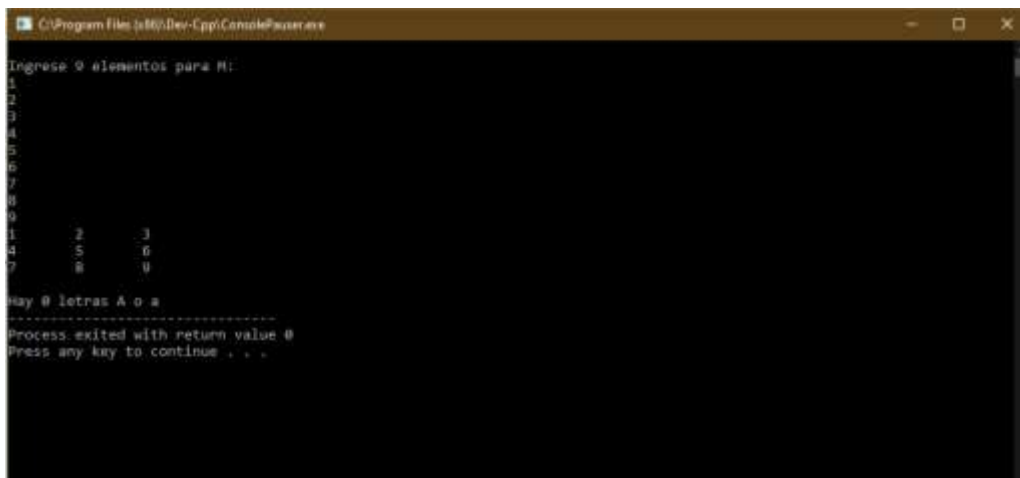
En C, las variables pueden ser declaradas en cuatro lugares del módulo del programa:

- Fuera de todas las funciones del programa, son las llamadas variables globales, accesibles desde cualquier parte del programa.
- Dentro de una función, son las llamadas variables locales, accesibles tan solo por la función en las que se declaran.
- Como parámetros a la función, accesibles de igual forma que si se declararan dentro de la función.

- Dentro de un bloque de código del programa, accesible tan solo dentro del bloque donde se declara. Esta forma de declaración puede interpretarse como una variable local del bloque donde se declara.

A continuación, se presentan algunos programas desarrollados en lenguaje C

```
1  #include<stdio.h>
2  int main() {
3      int a,b,c=0;
4      char M[3][3];
5      printf("\nIngrese 9 elementos para M: \n");
6      for(a=0;a<3;a++){
7          for(b=0;b<3;b++){
8              scanf("%c",&M[a][b]);
9              if(M[a][b]=='A' || M[a][b]=='a')
10                 c++;
11          }
12      }
13      for(a=0;a<3;a++){
14          for(b=0;b<3;b++){
15              printf("%c\t",M[a][b]);
16          }
17          printf("\n");
18      }
19      printf("\nHay %d letras A o a",c);
20      return 0;
21  }
```



```
C:\Program Files (x86)\Dev-Cpp\ConsolePase.exe
Ingrese 9 elementos para M:
1
2
3
4
5
6
7
8
9
1  2    3
4  5    6
7  8    9

Hay 0 letras A o a
Process exited with return value 0
Press any key to continue . . .
```

```

1  #include <stdio.h>
2  #include <stdlib.h> //biblioteca para borrar la pantalla
3  #define p printf
4  #define s scanf
5  int main(){
6      int f,c;
7      float m[5][5], suma[5];
8      p("Matriz 1\n");
9      for(f=0;f<=4;f++){
10         for(c=0;c<=4;c++){
11             p("Ingrese el el dato de la fila %d y de la columna %d\n",f+1,c+1);
12             s("%f",&m[f][c]);
13         }
14     }
15     for(f=0;f<=4;f++){
16         for(c=0;c<=4;c++){
17             p("%.3f\t",m[f][c]);
18         }
19         p("\n");
20     }
21     p("Suma de filas\n");
22     for(f=0;f<=4;f++){
23         for(c=0;c<=4;c++){
24             suma[c]+=m[c][f];
25         }
26     }
27     for(c=0;c<=4;c++){
28         p("%.3f\t",suma[c]);
29     }
30     p("\n");
31     return 0;
32 }

```

También de la forma

65. Leer dos matrices de 4 x 4 de tipo entero, almacenar las diagonales en vectores diferentes, obtener la suma de los diagonales –vectores- en otro vector. Mostrar cada arreglo en pantalla.

Ejemplo de 2X2:

M[2][2]	R[2][2]
3 2	5 6
4 5	7 9

Diagonal1[2]	Diagonal2[2]	Suma[2]
3	5	8
5	9	14

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

```
#include <stdlib.h> //biblioteca para borrar la pantalla
```

```
#define p printf
```

```
#define s scanf
```

```
int main(){
```

```

int f,c,m[4][4],m2[4][4],d[4],d2[4],suma[4];

p("Matriz 1\n");

for(f=0;f<=3;f++){

for(c=0;c<=3;c++){

p("Ingrese el el dato de la fila %d y de la columna %d\n",f+1,c+1);

s("%d",&m[f][c]);

if(f==c)

d[f]=m[f][c];

}

}

p("Matriz 2\n");

for(f=0;f<=3;f++){

for(c=0;c<=3;c++){

p("Ingrese el el dato de la fila %d y de la columna %d\n",f+1,c+1);

s("%d",&m2[f][c]);

if(f==c)

d2[f]=m2[f][c];

}

```



```
}
```

```
for(f=0;f<=3;f++){
```

```
    suma[f]=d[f]+d2[f];
```

```
}
```

```
p("Matriz 1\n");
```

```
for(f=0;f<=3;f++){
```

```
    for(c=0;c<=3;c++){
```

```
        p("%d\t",m[f][c]);
```

```
    }
```

```
p("\n");
```

```
}
```

```
p("Matriz 2\n");
```

```
for(f=0;f<=3;f++){
```

```
    for(c=0;c<=3;c++){
```

```
        p("%d\t",m2[f][c]);
```

```
    }
```

```
p("\n");
```

```
}
```

```
p("Diagonal 1\n");
```

```
for(f=0;f<=3;f++){
```

```
p("%d\t",d[f]);
```

```
}
```

```
p("\n");
```

```
p("Diagonal 2\n");
```

```
for(f=0;f<=3;f++){
```

```
p("%d\t",d2[f]);
```

```
}
```

```
p("\n");
```

```
p("Suma\n");
```

```
for(f=0;f<=3;f++){
```

```
p("%d\t",suma[f]);
```

```
}
```

```
p("\n");
```

```
return 0;
```

```
}
```