UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

ASIGNATURA: ESTRUCTURA DE DATOS Y ALGORITMOS

ACTIVIDAD #1 lunes

NOMBRE DEL ALUMNO: RIVERA VARGAS DONOVAN JOSHEP

APUNTADORES

APUNTADORES

Un **apuntador** es una variable **que** contiene la dirección de memoria de otra variable. Los **apuntadores** se utilizan para dar claridad y simplicidad a las operaciones a nivel de memoria.

Declaración de apuntadores

- Los apuntadores como cualquier otra variable deben de ser declarados antes de que puedan ser utilizados.
- El tipo de un apuntador lo proporciona implicitamente el tipo de la variable a la que apunta.
- Los apuntadores pueden ser declarados para apuntar a objetos de cualquier clase.
- La sintaxis general de declaración es:

  <tipo> * <variable>

- Ejemplos de declaraciones:

  int *contPtr, cont;
  float *res;
  unsugned int *nosigno;
  char *mensaje;

- La variable contPtr es del tipo apuntador a entero, (**int** *) y se lee ``contPtr es un apuntador a int'' o ``contPtr apunta a una variable entera''.

Los operadores de los apuntadores

1. Un operador de *dirección &*

- operador unario que regresa la direcci'on de su operando

- ejemplo:

prog33.c Primer ejemplo apuntadores

```
#include <stdio.h>
int main()

{
        int y;
        int *yPtr;

        y = 5;
        yPtr = &y;
        printf("Valor de y: %d \n",y);
        printf("Valor de yPtr: %d
\n",yPtr);
}
```

2. Un operador de indirección o de desreferencia

- regresa el valor del objeto hacia el cual su operando apunta, es decir un apuntador

prog34.c Segundo ejemplo apuntadores

```
#include <stdio.h>
main( )
{

        int x,y;
        int *py;
```

```
        y = 5;
        *py = y;
        x = *py + 5;
        printf("Valor de x: %d y de y: %d   \n",*py,x);
}
```

Apuntadores y argumentos funciones

- Pasando parámetros por valor

```
void permuta(x, y)
        int x,y;
{
        int temp;


        temp =x;
        x = y;
        y = x;
}
```

- Pasando parámetros por referencia

```
void permuta(px, py)

        int *px,*py;
{
        int temp;
        temp = *px;
        *px = *py;
        *py = *px;
}
```

- Programa incorrecto
  - El valor de n desplegado no corresponde al valor capturado, debido a que a scanf() se le debe pasar la dirección de una variable

```
int main( )
{
        int n;

        scanf("%d",n);
        printf("%d",n);
}
```

- Programa ``corregido''
  - El apuntador pn no apunta a nada, y scanf() intenta almacenar el valor capturado en la dirección almacenada en pn

```
#include <stdio.h>
int main( )
{
        int *pn;

        scanf("%d",pn);
        printf("%d",*pn);
}
```

- Programa que compila y corre
  - La dirección es pasada usando un parasitaje

```
#include <stdio.h>
int main( )
{
        int n;
        int *pn = &n;

        scanf("%d",pn);
        printf("%d",n);
}
```

- Programa Correcto
  - Programa que funciona seg

```
#include <stdio.h>
int main( )
{
        int n;

        scanf("%d",&n);
        printf("%d",n);
}
```