DOCUMENT_HEADING

9 files

(file list disabled)

t7\ejercicio1.c

```c
/*
  Implementa un programa que defina un puntero a un entero.

*/
#include <stdio.h>

int main() {
    int *puntero_entero;
    int entero = 10;

    // Se asigna la dirección de memoria del entero al puntero
    puntero_entero = &entero;

    printf("El valor del entero es: %d\n", entero);
    printf("La dirección de memoria del entero es: %p\n", &entero);
    printf("El valor del puntero es: %p\n", puntero_entero);

    return 0;
}
```

t7\ejercicio2.c

```c
/*
  Implementa un programa en C que muestre el uso de operaciones aritméticas
con punteros.
*/
#include <stdio.h>

int main() {
    int numeros[] = {10, 20, 30, 40, 50};
    int *puntero_numeros;

    // Se asigna el puntero a la primera posición del array
    puntero_numeros = numeros;

    printf("Los valores del array son: ");
    for(int i = 0; i < 5; i++) {
        printf("%d ", *(puntero_numeros + i));
    }

    printf("\nEl valor del primer elemento es: %d\n", *puntero_numeros);

    // Se realiza una operación aritmética con el puntero
    puntero_numeros += 2;

    printf("El valor del tercer elemento es: %d\n", *puntero_numeros);
```

```
    return 0;
}
```

t7\ejercicio3.c

```
/*
  Implementa un programa que muestre el uso de punteros y arrays.

*/
#include <stdio.h>

#define TAMANIO 5

int main() {
    int numeros[TAMANIO] = {10, 20, 30, 40, 50};
    int *puntero_numeros;

    // Se asigna el puntero a la primera posición del array
    puntero_numeros = numeros;

    printf("Los valores del array son: ");
    for(int i = 0; i < TAMANIO; i++) {
        printf("%d ", *(puntero_numeros + i));
    }

    printf("\nLa dirección de memoria del primer elemento es: %p\n",
puntero_numeros);
    printf("La dirección de memoria del segundo elemento es: %p\n",
puntero_numeros + 1);

    return 0;
}
```

t7\ejercicio4.c

```
/*
  Implementa un programa que muestre el uso de punteros a cadenas de
caracteres.

*/
#include <stdio.h>

int main() {
    char *puntero_cadena;
    char cadena[] = "Hola mundo!";

    // Se asigna el puntero a la primera posición de la cadena
    puntero_cadena = cadena;

    printf("El valor de la cadena es: %s\n", cadena);
    printf("El valor del puntero es: %p\n", puntero_cadena);
    printf("El primer caracter de la cadena es: %c\n", *puntero_cadena);
```

```
    // Se utiliza la aritmética de punteros para acceder a los caracteres de la cadena
    puntero_cadena += 4;

    printf("El quinto caracter de la cadena es: %c\n", *puntero_cadena);

    return 0;
}
```

t7\ejercicio5.c

```
/*
  Implementa un programa en C que asigne valores a un array usando punteros.

*/
#include <stdio.h>

#define TAMANIO 5

int main() {
    int numeros[TAMANIO];
    int *puntero_numeros;

    // Se asigna el puntero a la primera posición del array
    puntero_numeros = numeros;

    // Se utiliza un ciclo para asignar valores al array a través del puntero
    for(int i = 0; i < TAMANIO; i++) {
        *(puntero_numeros + i) = i * 10;
    }

    // Se imprime el contenido del array
    printf("Los valores del array son: ");
    for(int i = 0; i < TAMANIO; i++) {
        printf("%d ", numeros[i]);
    }

    return 0;
}
```

t7\ejercicio6.c

```
/*
  Implementa un programa en C que muestre el uso de matrices de punteros.

*/
#include <stdio.h>

#define FILAS 3
#define COLUMNAS 2

int main() {
    int numeros[FILAS][COLUMNAS] = {
        {10, 20},
```

```
        {30, 40},
        {50, 60}
    };

    int *punteros[FILAS];

    // Se asignan los punteros a las filas de la matriz
    for(int i = 0; i < FILAS; i++) {
        punteros[i] = numeros[i];
    }

    // Se imprimen los valores de la matriz utilizando los punteros
    printf("Los valores de la matriz son:\n");
    for(int i = 0; i < FILAS; i++) {
        for(int j = 0; j < COLUMNAS; j++) {
            printf("%d ", *(punteros[i] + j));
        }
        printf("\n");
    }

    return 0;
}
```

t7\ejercicio7.c

```
/*
  Implementa un programa en C que muestre el uso de indirecciones con
punteros.

*/
#include <stdio.h>

int main() {
    int numero = 10;
    int *puntero_numero;

    // Se asigna el puntero a la dirección de memoria de la variable numero
    puntero_numero = &numero;

    // Se utiliza la indirección para modificar el valor de la variable a
través del puntero
    *puntero_numero = 20;

    // Se imprime el valor de la variable utilizando la indirección a través
del puntero
    printf("El valor de la variable es: %d\n", *puntero_numero);

    return 0;
}
```

t7\ejercicio8.c

```
/*
```

```
  Implementa un programa en C defina un puntero a una estructura y se pase
como argumento a una función.

*/
#include <stdio.h>

// Se define la estructura
struct persona {
    char nombre[20];
    int edad;
};

// Se define la función que recibe un puntero a la estructura como argumento
void imprimir_persona(struct persona *puntero_persona) {
    printf("Nombre: %s\n", puntero_persona->nombre);
    printf("Edad: %d\n", puntero_persona->edad);
}

int main() {
    // Se declara una variable de la estructura
    struct persona p = {"Juan", 25};

    // Se declara un puntero a la estructura y se asigna a la dirección de la
variable
    struct persona *puntero_p = &p;

    // Se llama a la función y se pasa el puntero como argumento
    imprimir_persona(puntero_p);

    return 0;
}
```

t7\ejercicio9.c

```
/*
  Implementa un programa en lenguaje C que cree un puntero y se le asigne
memoria con la función malloc().

*/
#include <stdio.h>
#include <stdlib.h>

int main() {
    int *puntero_entero;
    int tamano = 5;

    // Se utiliza la función malloc para asignar memoria al puntero
    puntero_entero = (int*) malloc(tamano * sizeof(int));

    // Se verifica si se asignó correctamente la memoria
    if (puntero_entero == NULL) {
        printf("No se pudo asignar memoria\n");
        return 1;
    }

    // Se utiliza el puntero para asignar valores a los elementos del arreglo
    for(int i = 0; i < tamano; i++) {
```

```
        *(puntero_entero + i) = i * 10;
    }

    // Se imprime el contenido del arreglo a través del puntero
    printf("Los valores del arreglo son: ");
    for(int i = 0; i < tamano; i++) {
        printf("%d ", *(puntero_entero + i));
    }
    printf("\n");

    // Se libera la memoria asignada con la función free
    free(puntero_entero);

    return 0;
}
```