UNIVERSIDADE DA CORUÑA

FACULTADE DE INFORMÁTICA

Departamento de Ciencias da Computación e Tecnoloxías da Información

Programación II

Práctica 0a: Compilador y Punteros

Ejercicios:

1. Distinguir entre errores de compilación y errores de ejecución con punteros. Para ello asumir las siguientes declaraciones:

```
int x;
int* P1, * P2;
float* Q1, * Q2;
```

Construir un programa principal añadiendo de cada vez **sólo una** de las siguientes sentencias para probar si son correctas.

```
a) printf("%p",P1);
b) P1 = Q1;
c) if  (*P1 == NULL) { Q1 = Q2; }
d) scanf("%d",P1);
e) scanf("%d",P1);
   printf("El valor es %p", P1);
f) X = malloc(sizeof(int));
g) *Q1 = *P1
h) *P1 = 17;
   P1 = malloc(sizeof(int));
```

2. Con la misma declaración de variables del ejercicio anterior, compilar y ejecutar el siguiente código:

```
P1 = malloc(sizeof(int));
*P1 = 123;
printf("P1 es %d",*P1);
free(P1);
P1 = NULL;
printf("P1 es %d",*P1);
```

¿Por qué se produce un error de ejecución?

3. El siguiente programa maneja un tipo de dato tTemperatura como un registro con dos campos: alta y baja, que indican el valor de temperatura considerado como alto y bajo respectivamente. Comprobar su funcionamiento.

```
#include <stdio.h>

typedef struct {
    float alta;
    float baja;
} tTemperatura;

tTemperatura tmp;
float actual;


void registroTemp (float actual, tTemperatura *t)
{  if (actual > t->alta)
    t->alta = actual;
   else
       if (actual < t->baja)
           t->baja = actual;
}

void variaTemp (float actual)
{
    actual = 40;
}


int main() {
   tmp.alta = 30;
   tmp.baja = 15;
   printf("Temperatura inicial: %.2f alta, %.2f
baja",tmp.alta,tmp.baja);
   printf("Introduce temperatura actual:");    scanf("%f",&actual);
   registroTemp (actual,&tmp);
   printf("Los valores de temperatura ahora: %.2f alta, %.2f
baja",tmp.alta,tmp.baja);
   variaTemp(actual);
   printf("La temperatura actual no varía: %f \n",actual);
   return 0;
}
```

4. El siguiente programa maneja un tipo de dato `tEstudiante` como un registro con dos campos: `nombre` y `teléfono`, y un tipo de dato `pNodo` puntero a registro. El objetivo es:
   - Pedir por teclado los datos de dos estudiantes y almacenarlos a través del puntero.
   - Comparar los dos registros (campo a campo) y determinar si son o no iguales.

   Rellenar las operaciones `CrearNodo`, `MeterDatos` y `Comparar` de acuerdo a la descripción que las acompaña.

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

#define NULO NULL

     /* Definicion de tipos */


typedef struct {
    char nombre[100];
    int tlfno;
} tEstudiante;

typedef tEstudiante* pNodo;



void crearNodo(pNodo* p) {
…
}

void meterDatos (pNodo* d) {
/* Pide por teclado nombre y teléfono y los almacena en d */
…
}

bool comparar (pNodo p1, pNodo p2) {
/* Compara el contenido de las variables dinámicas asociadas */
/* a los punteros p1 y p2, y devuelve Verdadero (0) o Falso (1) */
…
}

int main () {
    pNodo p, q;

…
}
```