Informática I
Guía de Práctico
Uniones y estructuras

Martin Nievas
mnievas@frc.utn.edu.ar

30 de septiembre de 2019

Ejercicio 1

En los siguientes fragmentos de programa, determinar si son correctos o no. En caso de no serlo justificar.

1. _

```
union valores {
  char a;
  float b;
  double c;
};
union valores p = { 1.27 };
```

2. _

```
struct person {
  char lastName[15];
  char firstName[15];
  unsigned int age;
}
```

3. Suponga que estructura `carta` se ha definido con dos punteros para char (número y palo). Además, la variable c se ha definido para ser de tipo struct carta y la variable cPtr se ha definido para ser de tipo puntero a struct carta. A la variable cPtr se le ha asignado la dirección de c.

```
struct carta c;
cPtr = &c;
printf("%s\n", *cPtr->face);
```

Ejercicio 2

Indique si cada uno de los siguientes es verdadero o falso. Si es falso, explique por qué.

1. Las estructuras pueden contener variables de un solo tipo de datos.
2. Se pueden comparar dos uniones (usando ==) para determinar si son iguales.
3. El nombre de la etiqueta de una estructura es opcional.
4. Los miembros de diferentes estructuras deben tener nombres únicos.
5. La palabra clave typedef se usa para definir nuevos tipos de datos.
6. Las estructuras siempre se pasan a las funciones por referencia.
7. Las estructuras no pueden compararse utilizando operadores == y! =.

Ejercicio 3

Escriba el código para lograr cada uno de los siguientes:

1. Defina una estructura llamada parte que contenga la variable int sin signo partNumber y char array partName con valores que pueden tener hasta 25 caracteres (incluido el carácter nulo de terminación).
2. Definir Parte como sinónimo de la parte de tipo estructura.
3. Use Part para declarar que la variable a sea de tipo struct part, la matriz b [10] sea de tipo struct parte y la variable ptr para que sea de tipo puntero a struct part.
4. Lea un número de parte y un nombre de parte del teclado en los miembros individuales de la variable a.
5. Asigne los valores de miembro de la variable a al elemento 3 de la matriz b.
6. Asigne la dirección de la matriz b a la variable de puntero ptr.
7. Imprima los valores de miembros del elemento 3 de la matriz b usando la variable ptr y el operador de puntero de estructura para referirse a los miembros.

Ejercicio 4

Dada la estructura `personaje_t`, realizar un programa que solicite al usuario ingresar los datos necesarios para completar la estructura. Los miembros `escudo` y `sales` deben ser inicializados en 0. El miembro `vida` en 150,0. El miembro `rango` se debe generar en forma aleatoria, con un valor entre 0 y 100.

```
typedef struct {
  char apodo[20];
  int rango;
  float vida;
  float escudo;
  float sales;

  struct {
    char nombre[20];
    char apellido[20];
    int edad;
  } info_personal;

}personaje_t;
```

El programa debe cargar los miembros mediante dos formas: la primera en forma directa con la variable `personaje`, y en forma indirecta mediante el puntero `pPersonaje` según la siguiente definición:

```
personaje_t personaje, *pPersonaje;
pPersonaje = &personaje;
```

Acontinuación se adjunta un ejemplo de salida válido:

```
----------Cargar los elementos diréctamente----------
Ingrese su apodo: snow
Ingrese su nombre: martin
Ingrese su apellido: nievas
Ingrese su edad: 25
----------Datos----------
El apodo es: snow
Rango: 84
Vida: 150.000000
Escudo 0.000000
Sales: 0.000000
El nombre es: martin
El apellido es: nievas
----------Cargar los elementos mediante puntero----------
Ingrese su apodo: snow
Ingrese su nombre: martin
Ingrese su apellido: nievas
Ingrese su edad: 25
----------Datos----------
El apodo es: snow
Rango: 87
Vida: 150.000000
Escudo 0.000000
Sales: 0.000000
El nombre es: martin
El apellido es: nievas
```