ICF 122 - Fundamentos de Programación
Ingeniería Civil Informática

Unidad 6: Estructuras

Facultad de Ingeniería

Universidad Andrés Bello

Estructuras

- Una **estructura** es una colección de variables, no necesariamente del mismo tipo, que se referencian con un nombre común.
- Cada variable de la estructura se llama, elemento de la estructura.
- Sintaxis:

```
struct nombre_estructura{
  tipo_1 identificador_1;
  tipo_2 identificador_2;
  ...
  tipo_N identificador_N;
};
```

- Luego de crear la estructura, se pueden declarar variables del nuevo tipo de dato

```
struct nombre_estructura variable_estructura;
```

Estructuras

- Para acceder a algún elemento de la estructura, se utiliza el **operador miembro**, que es mediante el identificador punto:

```
variable_estructura.identificador_1
```

- Por ejemplo, para ingresar datos a la estructura:

```
variable_estructura.identificador_1 = valor_1;
```

- La declaración de una estructura se hace habitualmente fuera de cualquier función, puesto que el tipo de dato trasciende el ámbito de una función concreta.
- De todas formas, también se puede crear el tipo de dato dentro de la función, cuando ese tipo de dato no va a ser empleado más allá de esa función.

Estructuras

- Otro modo de generar la estructura y a la vez declarar las primeras variables de ese tipo, será la siguiente sintaxis:

```
struct nombre_estructura{
  tipo_1 identificador_1;
  tipo_2 identificador_2;
  ...
  tipo_N identificador_N;
}variable_1, ..., variable_N;
```

- Y así queda definido el identificador `nombre_estructura` y quedan declaradas las variables `variable_1`, ..., `variable_N`, que serán locales o globales según se hay hecho esta declaración en uno u otro ámbito.

Estructuras

Ejemplo

- Podemos necesitar definir un tipo de dato que podamos luego emplear para realizar operaciones en variable compleja. Esta estructura, que podríamos llamar `complejo`, tendría la siguiente forma:

```
struct complejo{
  double real;
  double imag;
};
```

- Y también podríamos definir una serie de operaciones, mediante funciones, por ejemplo, la suma, la resta y el producto de complejos:

```
struct complejo sumac(struct complejo, struct complejo);
struct complejo restc(struct complejo, struct complejo);
struct complejo prodc(struct complejo, struct complejo);
```

- Podemos notar que es un poco incómodo estar siempre escribiendo la palabra clave `struct` en cada declaración de un complejo. Una forma más cómoda para la creación de estructuras es mediante la combinación de la palabra `struct` y de la palabra `typedef`.

- La sintaxis de esa forma de creación es la siguiente:

```
typedef struct{
  tipo_1 identificador_1;
  tipo_2 identificador_2;
  ...
  tipo_N identificador_N;
} nombre_estructura;
```

- Y así, a partir de este momento, en cualquier lugar del ámbito de esta definición del nuevo tipo de dato, podremos crear variables con la siguiente sintaxis:

```
nombre_estructura nombre_variable;
```

- Volvamos a nuestro ejemplo, ahora la creación de `complejo` debe cambiar:

```
typedef struct{
  double real;
  double imag;
}complejo;
```

- Y la declaración de las funciones:

```
complejo sumac(complejo, complejo);
complejo restc(complejo, complejo);
complejo prodc(complejo, complejo);
```

Estructuras

Ejemplo

```
complejo sumac(complejo c1, complejo c2){
  c1.real += c2.real;
  c1.imag += c2.imag;
  return c1;
}

complejo restc(complejo c1, complejo c2){
  c1.real -= c2.real;
  c1.imag -= c2.imag;
  return c1;
}

complejo prodc(complejo c1, complejo c2){
  complejo S;
  S.real = c1.real + c2.real; - c1.imag * c2.imag;
  S.imag = c1.real + c2.imag + c1.imag * c2.real;
  return S;
}
```