Laboratorio de Computación II

Clase 03:
Estructuras

Estructuras

Un struct es una palabra reservada que nos permitirá generar un tipo de dato capaz de recrear una entidad en nuestra aplicación.

Alumno

Legajo
Apellido y nombres
Domicilio
Carrera
Nacionalidad

Entidad Alumno

```
struct Alumno{
  int legajo;
  char apellido[50];
  char nombre[50];
  char domicilio[120];
  char carrera[50];
  int cod_pais;
};
```

Estructura de Alumno

Variables de nuestra estructura

Como las estructuras que crearemos son tipos de datos. Podremos crear variables, arrays y punteros con ellas.

```
struct Alumno var, var2, var3;
```

```
struct Alumno lista[20], aprobados[15], ausentes[5];
```

```
struct Alumno *punt1, *punt2, *punt3;
```

Operador punto

Utilizaremos el punto para acceder a cada campo de una variable de tipo estructura.

Ejemplo

```
struct Alumno alu;
alu.legajo = 1000;
cin >> alu.cod_pais;
strcpy(alu.apellido, "Pérez");
cout << alu.apellido;
cout << alu.legajo;
```

```
struct Alumno{
  int legajo;
  char apellido[50];
  char nombre[50];
  char domicilio[120];
  char carrera[50];
  int cod_pais;
};
```

Estructura de Alumno

Operador flecha

Utilizaremos el operador flecha para acceder a cada campo de una variable de tipo estructura a través de un puntero.

Ejemplo

```
struct Alumno *alu, simple;
alu = &simple;
alu->legajo = 1000;
cin >> alu->cod_pais;
strcpy(alu->apellido, "Pérez");
cout << alu->apellido;
cout << alu->legajo;
```

```
struct Alumno{
  int legajo;
  char apellido[50];
  char nombre[50];
  char domicilio[120];
  char carrera[50];
  int cod_pais;
};
```

Estructura de Alumno

Operador []

Utilizaremos los corchetes para trabajar con vectores o con punteros a vectores. Luego el punto para acceder a los campos.

Ejemplo

```
struct Alumno alu[5];
alu[0].legajo = 1000;
cin >> alu[0].cod_pais;
strcpy(alu[0].apellido, "Pérez");
cout << alu[0].apellido;
cout << alu[0].legajo;
```

```
struct Alumno{
  int legajo;
  char apellido[50];
  char nombre[50];
  char domicilio[120];
  char carrera[50];
  int cod_pais;
};
```

Estructura de Alumno

Parámetros y valor devuelto desde funciones

Una variable de tipo estructura puede enviarse como parámetro o ser devuelta como valor de retorno desde funciones.

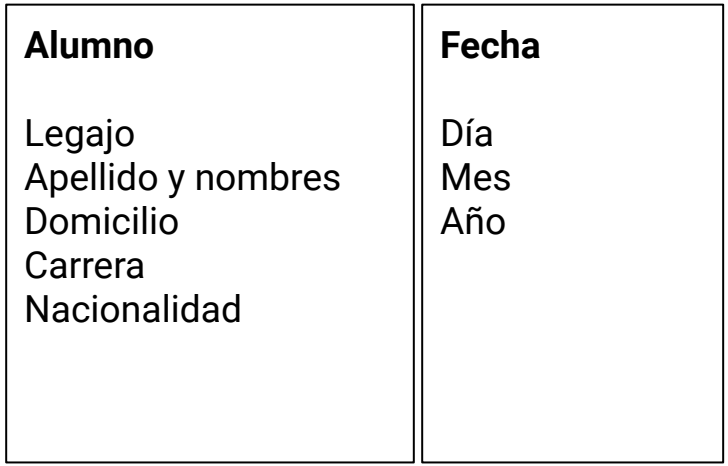
Ejemplos

```
void mostrar_alumno(Alumno reg){
  cout << reg.apellido << endl;
  cout << reg.nombre << endl;
}
```

```
Alumno cargar_alumno(){
  Alumno reg;
  cin >> reg.apellido;
  cin >> reg.nombre;
  return reg;
}
```

Estructura como atributo de una estructura

Una estructura puede estar compuesta por cualquier atributo de cualquier tipo de dato. Incluso de uno generado con una estructura.



Entidad Alumno | Entidad Fecha

```
struct Fecha{
  int dia;
  int mes;
  int anio;
};
```

Estructura de Fecha

```
struct Alumno{
  int legajo;
  Fecha nacimiento;
  char apellido[50];
  char nombre[50];
  char domicilio[120];
  char carrera[50];
  int cod_pais;
};
```

Estructura de Alumno

Estructura como atributo de una estructura

Una estructura puede estar compuesta por cualquier atributo de cualquier tipo de dato. Incluso de uno generado con una estructura.

```
Alumno cargar_alumno(){
  Alumno reg;
  cin >> reg.apellido;
  cin >> reg.nombre;
  cin >> reg.nacimiento.dia;
  cin >> reg.nacimiento.mes;
  cin >> reg.nacimiento.anio;
  return reg;
}
```

Actividad

Hacer un programa que permita cargar los 6 participantes a un torneo de pesca y listarlos por pantalla.

Participantes

Código de participante
Apellidos
Nombres
Género
Fecha de nacimiento

Restricciones

- El Código de participante debe ser un número entre 1000 y 9999.
- El Código de participante no puede repetirse en la lista.
- El Género debe ser un carácter 'F', 'M', o 'X'.
- El participante debe tener 18 años o más para poder participar.