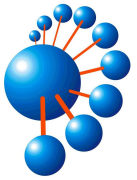
Programación I
2021-2

Clase 14
Estructuras y unión - parte 2

Universidad de Concepción

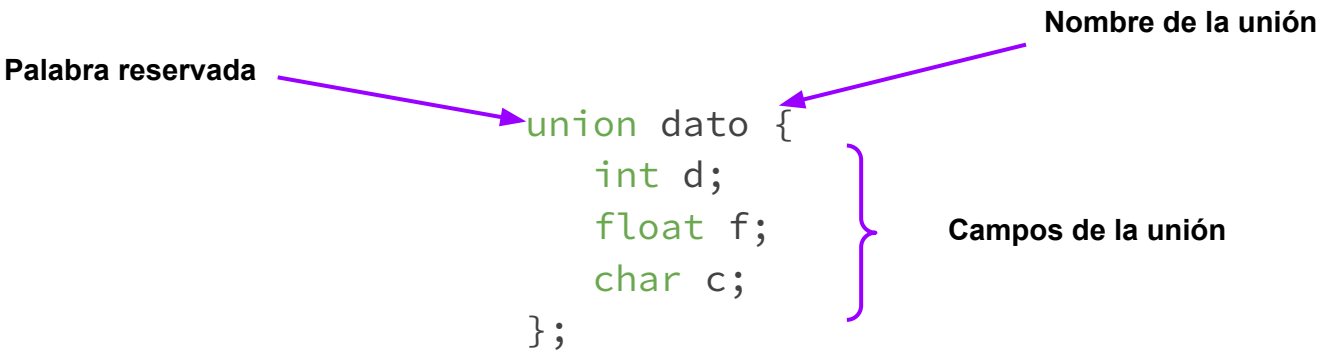
José Fuentes - jfuentess@inf.udec.cl

Departamento de Ingeniería Informática y Ciencias de la Computación



Unión (union)

Una colección de variables relacionadas, agrupadas bajo un mismo nombre. Similar a *struct*, pero sólo en una unión sólo un campo puede tener un valor válido a la vez.



Uniones proveen una manera de manipular diferentes tipos de datos en una misma área de memoria.

Union: en contexto

```c
#include <stdio.h>
#include <stdlib.h>

union dato {
   int d;
   float f;
   char c;
   char s[8];
};

void print_dato(union dato v) {
   printf("%d, %f, %c, %s\n", v.d, v.f, v.c, v.s);
}

int main() {
   union dato var;
   var.d = 103;
   print_dato(v);
   printf("Tamaño: %ld bytes\n", sizeof(union dato)); // 8 bytes
   return 0;
}
```

Acceso a los campos de la union

Declaración de una variable de tipo union dato

Ver: union.c

Enumeraciones: enum

Palabra reservada

Nombre del enum

Campos de enum (Todos constantes)

Declaración de variables enum

```c
#include <stdio.h>
#include <stdlib.h>

enum estado {ACTIVO, INACTIVO};
enum rating {EXCELENTE=5, BUENA=4, NORMAL=3, BAJA=2, MALA=1};
enum semana {LU, MA, MI, JU=10, VI, SA, DO};

int main() {
   enum estado e_actual;
   enum rating calidad;
   enum semana dia;

   e_actual = INACTIVO;
   calidad = NORMAL;
   dia = LU;

   printf("%d %d\n", ACTIVO, INACTIVO);
   printf("%d %d %d %d %d\n", EXCELENTE, BUENA, NORMAL, BAJA, MALA);
   printf("%d %d %d %d %d %d %d\n", LU, MA, MI, JU, VI, SA, DO);

   return 0;
}
```

Ver: enum.c

Union vs Struct

```c
typedef union u_dato t_u;

union u_dato {
   int d;
   float f;
   char c;
   char s[8];
};

printf("Tamaño: %ld bytes\n",
     sizeof(t_u)); // 8 bytes
```

```c
typedef struct s_dato t_s;

struct s_dato {
   int d;
   float f;
   char c;
   char s[8];
};

printf("Tamaño: %ld bytes\n",
     sizeof(t_s)); // 20 bytes
```

Ver: cmp_union_struct.c

Struct: listas enlazadas

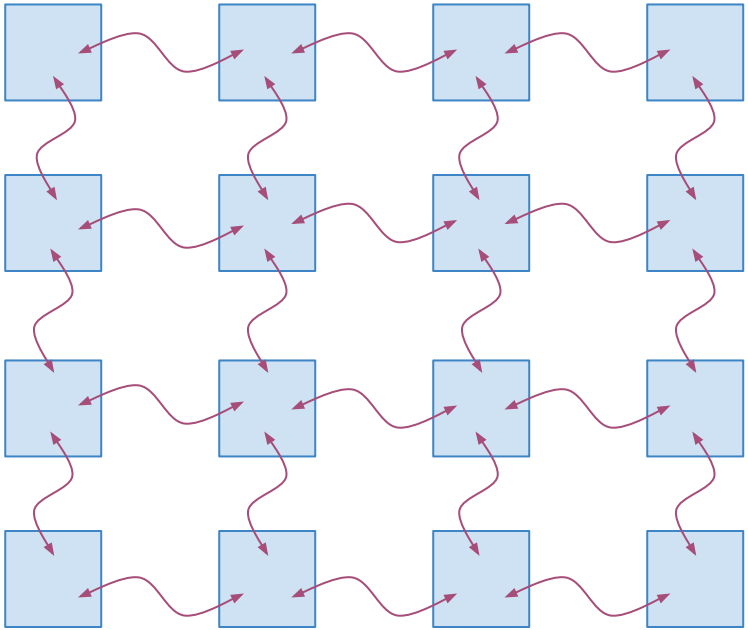
```c
struct nodo {
   int valor;
   struct nodo sig;
};
```


```c
int main() {
   struct nodo nodo1;
   struct nodo nodo2;

   nodo1.valor=10;
   nodo1.sig = &nodo2;

   nodo2.valor=15;
   nodo2.sig = NULL;

   return 0;
}
```

Ver: lista_simple.c

¿Cómo implementar este patrón?

Opción 1: `arreglos 2D`
Ventajas vs desventajas



Opción 2: `struct + punteros`
Ventajas vs desventajas

¡A practicar!

Ejemplo 1:
lista_completa.c

Ejemplo 2:
tienda_juegos.c

Ejemplo 3:
lista_doble.c