Unión

Unión

Una unión es un tipo de datos derivado, como una estructura, con miembros

que comparten el mismo espacio de almacenamiento. Una variable de tipo unión puede contener (en momentos diferentes) objetos de diferentes tipos y tamaños.

Las uniones proporcionan una forma de manipular diferentes clases de datos

dentro de una sola área de almacenamiento.

En cualquier momento una unión puede contener un máximo de un objeto

debido a que los miembros de una unión comparten el espacio de almacenamiento

Unión

Una unión se declara con el mismo formato de una struct. Primero declaramos el tipo unión y luego declaramos variables de ese tipo.

Test:
https://replit.com/join/qfxxkybqiz-maximo-arielari

```
#include <iostream>
using namespace std;
typedef char st40[40];
union persona{
        st40 nombre;
        int edad;
        char genero;
};
persona persona1;
int main() {
  cout << "Uniones \n";
  cout<<"\n Tamaño de la Union
:"<<sizeof(persona1)<<"Bytes";

}
```

Comparamos Struct vs Union

```
typedef char stnom[40];
typedef char stmail[200];
union persona{
        stnom nombre;
        int edad;
        stmail email;
};
```

Test: https://replit.com/join/lzcmftmyhh-maximo-arielari

```
typedef char stnom[40];
typedef char stmail[200];


struct personb{
        stnom nombre;
        int edad;
        stmail email;
};
```

Usabilidad

El uso debe ser consistente con los valores almacenados en cada caso, la cuestión es que esté queda bajo responsabilidad del programador (esta es la razón principal para que se desaconseje su uso salvo caso de necesidad insoslayable). Si se lee un dato de tipo distinto al que se escribió, el resultado obtenido es dependiente de la implementación.

Unión

Estas Variables comparten la misma sección de memoria, fueron constituidas para ahorrar espacio en memoria. La prioridad para soportar esta información es la última transferencia dada. De este modo determinamos que esta estructura no soporta la información simultánea.

Test : https://replit.com/join/dxqxddvzgh-maximo-arielari

Enumeraciones

Enumeraciones

Un enum es un tipo definido por el usuario con constantes de nombre de tipo entero.

enum dato{ enumerador1, enumerador2, enumerador3 }

enum

```
#include <iostream>

using namespace std;

enum week { Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday };

int main(){

    week today;

    today = Wednesday;

    cout << "Day " << today+1;

        return 0  }
```

Test: https://replit.com/join/lzgluizapq-maximo-arielari

enum

Otra posible implementación:

TEST:
https://replit.com/join/cktkskwyzl-maximo-arielari

```
#include <iostream>
using namespace std;
enum Boolean{
  FALSE, TRUE
};
Boolean vocal(char *s);
int main() {
  char t;
  cout << "Ingrese un Caracter : \n";
  cin>>t;
  if(vocal(&t)){cout<<"Es una Vocal";}
    else{ cout<<"Consonante"; }}
Boolean vocal(char *s){
  switch(*s){
    case 'a': return TRUE;
    case 'e': return TRUE;
    case 'i': return TRUE;
    case 'o': return TRUE;
    case 'u': return TRUE;
    defaul : return FALSE;
    }
}
```

Visibilidad

Las estructuras se han definido para compartir su información en el ámbito de la programación estructural o funcional.

La visibilidad de las mismas es de ámbito público.

Bibliografía

https://www.zator.com/Cpp/E4_7.htm#:~:text=Las%20uniones%20C%2B%2B%20son%20un,los%20posibles%2C%20al%20mismo%20tiempo.

https://www.fing.edu.uy/tecnoinf/mvd/cursos/eda/material/teo/EDA-teorico13.pdf

Programación en C++ autor: Luis Joyanes Aguilar.