UNION

lunedì 19 dicembre 2022 22:34

UNION

È una definizione simile alla struct perché all'interno io posso definire vari membri, ma all'interno della variabile di tipo unione può esistere un solo membro per volta, vuol dire che io sto riservando un'area di memoria e ci vado a scrivere un solo tipo per volta;

- L'accesso ai membri delle unioni segue le stesse regole delle struct
  - operatore punto per le variabili di tipo unione
  - operatore freccia per i puntatori alle variabili di tipo unione

Nel caso in cui ho una funzione che deve accettare più tipi e non voglio definire tante funzioni con nomi diversi per accettare tipi differenti, allora posso racchiudere i tipi differenti all'interno di un'unione e, a seconda dell'invocazione che farò, passerò esattamente un tipo specifico;

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

typedef enum type
{
    INT,
    FLOAT,
    DOUBLE
} type_t;
typedef union operand
{
    int i;
    float f;
    double d;
} operand_t;
```

insieme di tipi di dato

uno per volta

occupa un double in memoria! 8 byte. Quello maggiore. se scrivo un float o int, scrivo solo 4 byte!

```c
int main(void)
{
    operand_t
        op11 = { .i = 3 },
        op12 = { .i = 5 },
        op21 = { .f = 2.18f },
        op22 = { .f = 8.79f },
        op31 = { .d = 1.53 },
        op32 = { .d = 4.71 };
    sum(INT, op11, op12);
    sum(FLOAT, op21, op22);
    sum(DOUBLE, op31, op32);
    printf("\nsizeof(int) = %zu\n", sizeof(int));
    printf("sizeof(float) = %zu\n", sizeof(float));
    printf("sizeof(double) = %zu\n", sizeof(double));
    printf("sizeof(operand_t) = %zu\n", sizeof(operand_t));
    return 0;
}
```

int → 4, float → 4, double → 8, operand_t → 8

```c
void sum(type_t t, operand_t op1, operand_t op2)
{
    switch(t)
    {
    case INT:
        printf("%d + %d = %d\n", op1.i, op2.i, op1.i + op2.i); break;
    case FLOAT:
        printf("%.02f + %.02f = %.02f\n", op1.f, op2.f, op1.f + op2.f); break;
    case DOUBLE:
        printf("%.02lf + %.02lf = %.02lf\n", op1.d, op2.d, op1.d + op2.d); break;
    default:
        printf("Unexpected operand types.\n");
    }
}
```

somma fra interi

somma fra float

somma fra double

OUTPUT

```
3 + 5 = 8
2.18 + 8.79 = 10.97
1.53 + 4.71 = 6.24

sizeof(int) = 4
sizeof(float) = 4
sizeof(double) = 8
sizeof(operand_t) = 8
```