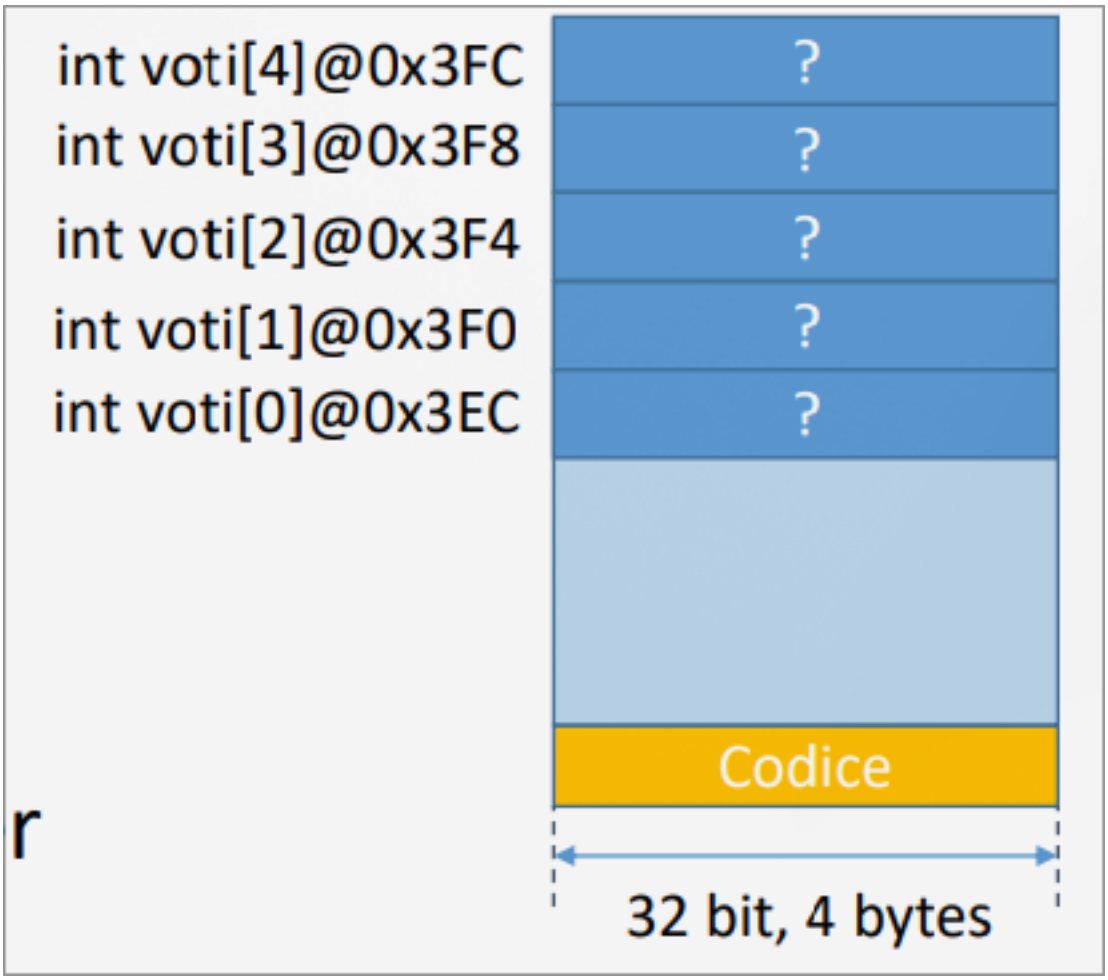
Contents

Array

Alloca dati in indirizzi contigui.

Il primo elemento dell'array è sempre quello all'indirizzo più basso dell'allocazione.



Usiamo size_t come tipo dell'indice nel for La dimensione dell'array è un #define

Tipo size_t

Lo specificatore per size_t per scanf e printf è %z.

Buffer overflow

Accade quando proviamo a leggere/modificare una cella dell'array che non esiste(ad esempio indice 5 su array[4])

Possono succedere 3 cose:

- Vengono letti dati sbagliati
- Il programma va in segmentation fault
- Vengono modificati dei dati nel programma

VLA

Dimensione dell'array decisa a runtime

Ricerca di elementi nell'array

Esistono algoritmi diverse per problemi diverse che hanno una complessità diversa.

Ricerca lineare

Dato un array di lunghezza lenA, possiamo scrivere l'algoritmo che ricerchi un determinato valore e ne restituisca l'indice.

Una soluzione corretta a livello algoritmico sarebbe:

```
int main(void) {
    //acquisizione lenA
    //allocazione i, acquisizione dati in a
    int key = 18;
    size_t i;
    for (i = 0; i < lenA; i++) {
        if (a[i] == key) {
            break;
        }
    }
    printf ("Elemento trovato in posizione %d\n", i);
}
```

Tuttavia viene usato un break nel ciclo e quindi è completamente sbagliata.

Questa è la versione corretta:

```
int main(void) {
    //acquisizione lenA
    //allocazione i, acquisizione dati in a
    int key = 18;
    size_t pos = lenA;
    for (size_ i = 0; i < lenA && pos == lenA ; i++) {
        if (a[i] == key) {
            pos = i;
        }
    }
    printf ("Elemento trovato in posizione %d\n", pos);
}
```

Filtraggio con ricerca lineare

Dato un array a e b, vogliamo filtrare gli elementi di a per una certa condizione e pusharli in b.

Usiamo quindi due indici nello stesso for.

Bubble Sort

Tutto si basa sullo scambiare le varibile se `cv[i] > v[i+1]`, finchè c'è questa condizione, la variabile sentinella viene messa come true.

```
#define SIZE_A 3

int main(void) {
    int v[] = {6, 2, -8};
    bool eseguiCiclo = true;
    while (eseguiciclo == true) {
        eseguiCiclo = false;
        for (size_t i = 0;i < SIZE_A-1; i++) {
            if (v[i] > v[i+1]) {
                // scambia v[i] e v[i+1]
                eseguiCiclo = true;
            }
        } // end for
    } // end while
}
```

SIZE__A-1 evita buffer overflow