

## RECORD DI ATTIVAZIONE

venerdì 2 dicembre 2022 22:52

# RECORD DI ATTIVAZIONE

LE VARIABILI LOCALI O AUTOMATICHE, OCCUPANO MEMORIA ALL'INTERNO DELLO STACK.

## COME VIENE GESTITO LO STACK DALLE FUNZIONI IN C.

NOI POSSIAMO CAPIRE, IN QUALE INVOCAZIONE DI UNA FUNZIONE CI TROVIAMO, PERCHÉ OGNI FUNZIONE CREA UN RECORD DI ATTIVAZIONE CHE DESCRIVE, QUANDO È COME LA FUNZIONE, È STATA ATTIVATA!

INT F ( VOID )

{     INT x;

    F()

    }

Mi DEVO RICORDARE CHE CI SONO PIÙ F ATTIVATE!  
DEVO DISTINGUERE CIASCUNA DI QUESTE INVOCAZIONI L'UNA DALL'ALTRA.

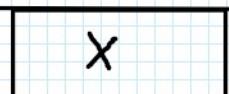
PERCHÉ SE DENTRO HO UNA VARIABILE LOCALE CHE DIPENDE DALL'ATTIVAZIONE, IL VALORE DI X DI UNA CERTA ATTIVAZIONE DI F DEVE ESSERE

DIVERSO DAL VALORE DI X DI UN'ALTRA ATTIVAZIONE!

AVRÒ UNA VARIABILE X PER OGNI INVOCAZIONE DI F().

DEVO DISTINGUERE LE VARI AREE DI MEMORIA IN CUI LE ISTANZE DI X VENGONO MEMORIZZATE.

OGNI VOLTA CHE LA MIA FUNZIONE VIENE INVOCATA, IO RISERVO UN PO' DI MEMORIA PER LA PORZIONE DI X DI QUESTA INVOCAZIONE; POI RICHIAMERO LA MIA FUNZIONE F() E LO STACK CRESCERÀ ALL'INDIETRO, DECREMENTANDO RSP;



OGNI VOLTA CHE UNA FUNZIONE PRENDE IL CONTROLLO, DEVE RISERVARE SULLA CIMA DELLO STACK UN PO' DI SPAZIO, PER TUTTE LE VARIABILI LOCALI CHE VERRANNO OSPITATE IN QUELLA FUNZIONE.

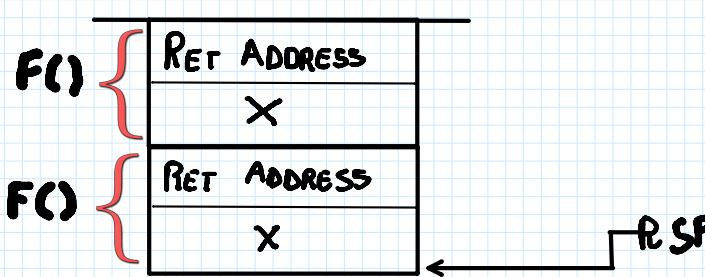
AREA DI MEMORIA ASSOCIATA ALL'INVOCAZIONE DI UNA DETERMINATA FUNZIONE.

MA QUANDO LA FUNZIONE TERMINA MI DEVO RICORDARE CHI È IL CHIAMANTE.

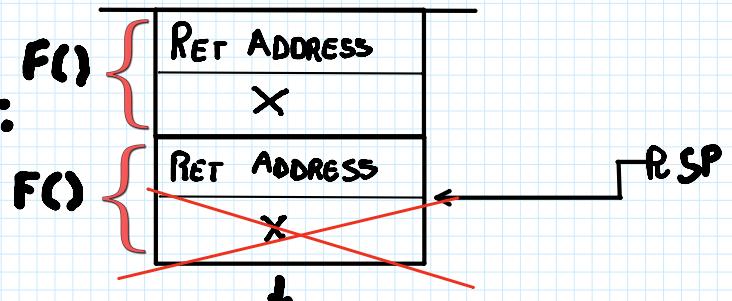
QUINDI NELLO STACK NON DEVO SOLTANTO RISERVARE LO SPAZIO PER LE MIE VARIABILI LOCALI, MA DEVO MANTENERE ANCHE L'INDIRIZZO DI RITORNO;

QUINDI SE IO INVOCO PER LA PRIMA VOLTA LA FUNZIONE F, IO DEVO RESTITUIRE IL CONTROLLO AL CHIAMANTE CHE AVEVA CHIAMATO F PER LA PRIMA VOLTA;

Poi F() invoca se stessa e io dovrò scrivere sulla cima dello stack un altro indirizzo di ritorno.



SE VOGLIO FAR RITORNARE IL VALORE:



Record invalidato logicamente

SOTTO RSP (L'INDIRIZZO contenuto da RSP) È LOGICAMENTE NON valido;

PERÒ IN MEMORIA LO LEGGO IL CONTENUTO DI X FINO A QUANDO NON RIMETTO QUALcosa SULLO STACK.

IL VECCHIO VALORE DI X È ANCORA LEGGIBILE, MA SE LO LEGGO SO ACCENDENDO AD UN VALORE LOGICAMENTE NON PIÙ VALIDO: undefined behaviour;

TUTTO CIÒ CHE È SOTTO LO SP È LOGICAMENTE INVALIDATO.

- quando chiama una funzione, si crea per essa un record di attivazione nel punto più basso libero dello stack;
- questo record di attivazione è una zona di memoria che contiene i suoi parametri formali e le sue variabili locali;
- quando la funzione termina, il record di attivazione viene cancellato dallo stack, liberando quindi la memoria per successivi record di attivazione.

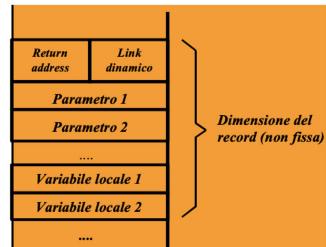
#### RECORD DI ATTIVAZIONE

- Rappresenta il "mondo della funzione": *nasce e muore con essa*
  - è creato al momento della invocazione di una funzione
  - permane per tutto il tempo in cui la funzione è in esecuzione
  - è distrutto (*deallocato*) al termine dell'esecuzione della funzione stessa.
- Ad ogni chiamata di funzione viene *creato un nuovo record, specifico per quella chiamata di quella funzione*
- La dimensione del record di attivazione
  - varia da una funzione all'altra
  - *per una data funzione, è fissa e calcolabile a priori*

#### RECORD DI ATTIVAZIONE

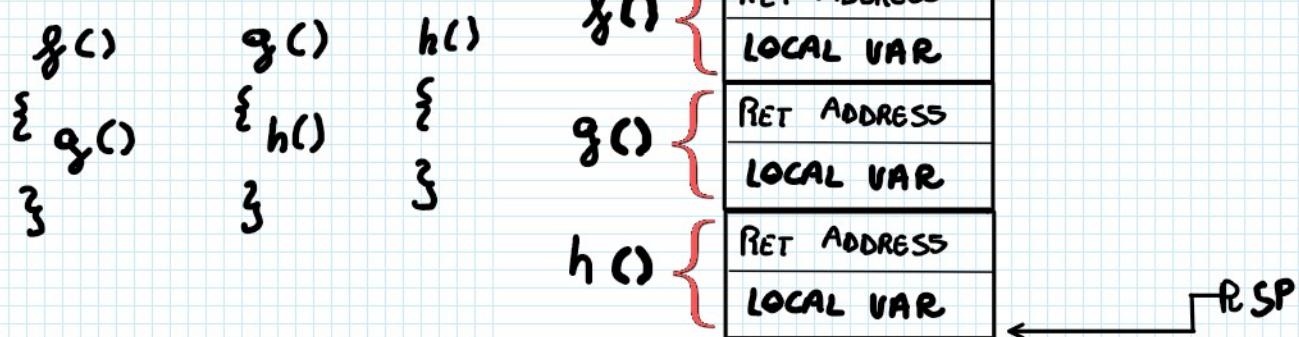
- Funzioni che chiamano altre funzioni danno luogo a una sequenza di record di attivazione
  - allocati secondo l'ordine delle chiamate
  - deallocati in ordine inverso
- La sequenza dei link dinamici costituisce la cosiddetta catena dinamica, che rappresenta la storia delle attivazioni ("chi ha chiamato chi")

#### RECORD DI ATTIVAZIONE



IL MAIN HA UN SUO RECORD DI ATTIVAZIONE;

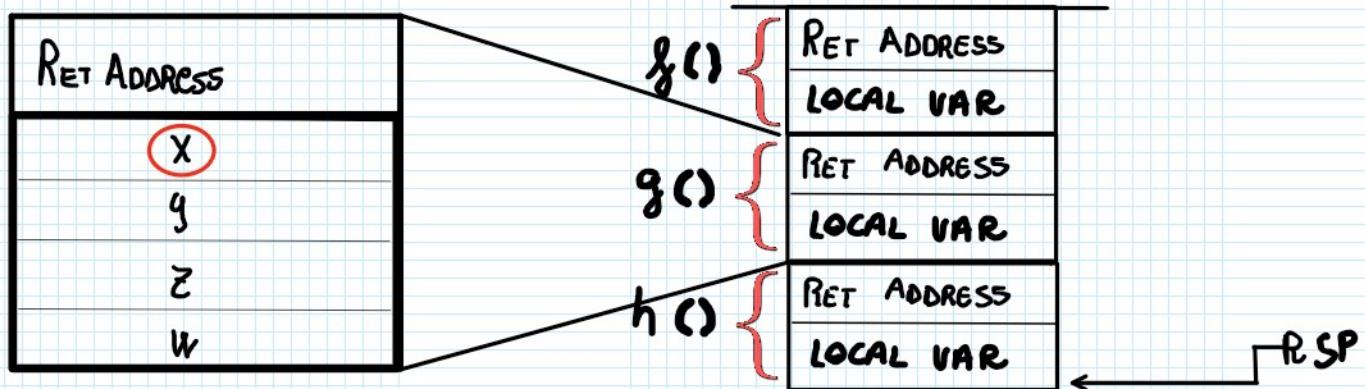
PROVIAMO:



IO VOGLIO POTER ACCEDERE ALLE MIE VARIABILI LOCALI E DEVO SAPERE QUANTO È GRANDE QUESTO RECORD DI ATTIVAZIONE E A QUALE SPIAZZAMENTO, RISPETTO ALLA BASE O ALLA CIMA DI QUESTO RECORD DI ATTIVAZIONE, SI TROVA UNA DETERMINATA VARIABILE.



ZOOM SU UN SINGOLO RECORD DI ATTIVAZIONE!



DEVO POTER SAPERE qual'è l'indirizzo di x;

SI UTILIZZANO DUE REGISTRI DI PROCESSORE:

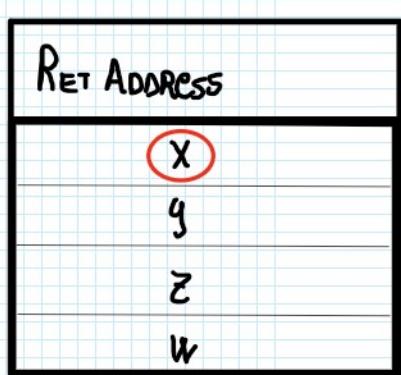
- **STACK POINTER (RSP)**: DEFINISCE IN OGNI Istanze LA CIMA DELLO STACK.
- **FRAME POINTER (FP)**: DEFINISCE LA BASE DEL RECORD DI ATTIVAZIONE.

È BEN evidente che se ci dovessero essere altre allocazioni di variabili nello stack, lo stack crescerebbe, andando a diminuire RSP.

SE io definisco l'indirizzo di x A PARTIRE dall'RSP, lo spiazzamento puo cambiare durante l'esecuzione della mia funzione; **QUESTA COSA È SCOMODA!**

Frame pointer

Esecuzione della mia funzione; **cosa succede?**



Frame pointer

IL FRAME POINTER INVECE È COSTANTE. DEFINISCE LA BASE DEL MIO RECORD DI ATTIVAZIONE, SEMPRE.

X si troverà sempre ad un valore di FP - h

↳ non dipende dall'RSP

Tutte le variabili locali della mia funzione avranno uno spiazzamento costante dall'FP, per tutta questa la vita della funzione;

↓                    ↓  
FINCHÉ LA STO ESEGUENDO