Corso di Laurea in Informatica

RELAZIONE

elaborato

ASSEMBLY

A cura di:

GIRARDELLO EMANUELE
VR500077

TESSERA MATTEO
VR502611

A.A 2023/2024

Relazione elaborato Assembly

INDICE

PROGETTO SISTEMA PRODUTTIVO

1.1 - DESCRIZIONE 1
1.2 - SPECIFICHE PRODOTTI 1
1.3 - CALCOLO PENALITÀ 2

STRUTTURA DEL CODICE

2.1 - STRUTTURE DATI 3
2.2 - FUNZIONI UTILIZZATE 3

LOGICA DEL CODICE

3.1 - DESCRIZIONE DEL SOFTWARE DI PIANIFICAZIONE 5
3.2 - GESTIONE DEI PARAMETRI E LETTURA DEL FILE 5
3.3 - STRUTTURAZIONE DEI DATI 5
3.4 - MENU DI SELEZIONE DELL'ALGORITMO 6
3.5 - ALGORITMI DI PIANIFICAZIONE 6
3.6 - OUTPUT DELLA PIANIFICAZIONE 6
3.7 - RIPETIZIONE DEL PROCESSO 7

ESEMPIO STACK

4.1 - DESCRIZIONE ESEMPIO 8

SCELTE PROGETTUALI

5.1 - DESCRIZIONE SCELTE PROGETTUALI 9

TEST EFFETTUATI

6.1 - DESCRIZIONE TEST 10

Relazione elaborato Assembly

PROGETTO SISTEMA PRODUTTIVO

1.1 DESCRIZIONE

Il progetto consiste nella realizzazione di un software in **assembly AT&T** per la pianificazione delle attività di un **sistema produttivo.**
Il sistema produttivo ha la capacità di produrre diversi tipi di prodotti, ma solamente uno alla volta.
La produzione è suddivisa in **slot temporali** e durante ogni slot solo un prodotto può essere in produzione.

1.2 SPECIFICHE PRODOTTI

Tutti i prodotti (massimo 10) che devono essere messi in produzione, dovranno essere contenuti in un file .txt.
Ogni prodotto dovrà essere descritto come segue:

identificativo, durata, scadenza, priorità

Identificativo:

- è un codice che identifica il prodotto da produrre.
- Range: 1 - 127

Durata:

- è il numero di slot temporali necessari per completare il prodotto.
- Range: 1-10 slot temporali.

Scadenza:

- è il tempo massimo, espresso come numero di unità di tempo, entro cui il prodotto dovrà essere completato.
- Range: 1 - 100 unità di tempo

Priorità:

- è un valore che specifica la priorità del prodotto.
  Questo valore sarà necessario anche per calcolare la penalità che l'azienda dovrà pagare per ogni unità di tempo necessaria a completare il prodotto oltre la scadenza.
- Range: 1 (priorità minima) - 5 (priorità massima)

1.3 CALCOLO PENALITÀ

La penalità che l'azienda dovrà pagare per ogni unità di tempo necessaria a completare il prodotto oltre la scadenza sarà calcolata in base alla priorità del prodotto.

Esempio:
Se il prodotto: **identificativo**: 4; **Durata**: 10; **Scadenza**: 25; **Priorità**: 4 venisse messo in produzione all'unità di tempo 21, il sistema completerebbe la sua produzione al tempo 30, con 5 unità di tempo di ritardo rispetto alla scadenza richiesta, l'azienda dovrebbe pagare una penalità di 5 * 4 = 20 Euro.

STRUTTURA CODICE

2.1 STRUTTURE DATI

Il codice è interamente contenuto nel file sorgente, non vengono utilizzate funzioni e non è diviso in più file
Il codice è suddiviso come segue:

- .section .bss: sezione standard assembly in cui vengono dichiarate le variabili globali e statiche che vengono inizializzate a zero o che non sono inizializzate.
- .section .data: sezione standard assembly in cui vengono dichiarate le variabili globali e statiche che vengono inizializzate con valori specifici.
  Ad esempio, qui dichiariamo le variabili di tipo stringa che utilizziamo per la stampa del menu, degli errori o di varie stringhe che abbiamo deciso di implementare nel nostro software.
- _start indica l'inizio del codice, abbiamo diviso il nostro codice in più etichette, ognuna delle quali implementa una certa funzionalità.

2.2 LE PARTI PRINCIPALI DEL PROGRAMMA E LA LORO FUNZIONE

- _start: preleva i parametri dati dalla linea di comando,
- estrae dalla pila, l'indirizzo che punt alla stringa contenete il primo parametro, ovvero il nome del file che intendiamo leggere.
- leggichar: estrae i caratteri presenti nel file .txt. finchè non incontra una virgola, quindi traduce i caratteri letti in un singolo valore intero
- pusha: carica il valore intero sullo stack (pila).

- **checkValori**: controlla se i valori estratti rispettano i requisiti specificati.
- **menu**: stampa il menu principale e richiede un input all'utente, il quale dovrà inserire un numero che corrisponderà ai seguenti comandi:
  - 0 per uscire
  - 1 per utilizzare l'algoritmo EDF
  - 2 per utilizzare l'algoritmo HPF
- **edf**: vengono inseriti in cima alla pila gli indirizzi che puntano ai parametri "scadenza" di ciascun ordine già presente nella pila. Ad esempio, se ci sono 10 ordini, vengono caricati 10 indirizzi che puntano rispettivamente alle scadenze dell'ordine 1, dell'ordine 2, e così via
- **hpf**: come edf, carica in pila gli indirizzi relativi alle priorità di ciascun ordine.
- **bubblesort**: implementa l'algoritmo di ordinamento Bubble Sort. e ordina gli indirizzi inseriti in cima alla pila seguendo le indicazioni dell'algoritmo scelto.
- **output hpf/edf**: stampa la stringa "Pianificazione HPF/EDF" in base all'algoritmo scelto.
- **output id e inizio**: stampa gli ID e l'inizio produzione di ciascun prodotto.
- **conclus / stampa_conclus**: stampa a quale unità di tempo si conclude la produzione.
- **calcPenalty / stampaPenalty**: stampa la penalità che l'utente dovrà pagare in termini di euro per i prodotti completati oltre la scadenza.
- **errori**: stampa stringhe di errore in base al problema riscontrato.

LOGICA CODICE

3.1 DESCRIZIONE DEL SOFTWARE DI PIANIFICAZIONE

Il nostro software è progettato per gestire e calcolare efficacemente la pianificazione della produzione basata su ordini specificati in file .txt.
Quando l'utente avvia il Makefile, vengono creati i file .obj e l'eseguibile pianificatore, salvati rispettivamente nelle cartelle obj e bin.
Nella cartella ordini sono presenti i file .txt relativi agli ordini su cui il software eseguirà la produzione.

3.2 GESTIONE DEI PARAMETRI E LETTURA DEL FILE

All'avvio, il software gestisce il passaggio del parametro relativo al percorso del file e del file .txt. Utilizzando una prima etichetta, il programma esegue una syscall per aprire il file e leggerne il contenuto.
I valori letti vengono suddivisi escludendo le virgole e viene effettuato un controllo per verificare che gli ordini siano conformi ai range previsti dalla specifica.
Se il file contiene valori al di fuori del range consentito, il software stampa un errore: **"Error: File con valori fuori dal range consentito".**

3.3 STRUTTURAZIONE DEI DATI

La strutturazione dei dati nel nostro programma si basa principalmente sull'uso della pila come struttura dati. La pila + risultata particolarmente adatta, in quanto ha permesso di salvare facilmente i parametri contenuti nel file in maniera ordinata.
Inoltre ha reso semplice e conveniente l'accesso ai dati (aggiungendo un offset al registro ESP). (vedi 4.1)

3.4 MENU DI SELEZIONE DELL'ALGORITMO

Dopo aver salvato i valori degli ordini nella pila, viene visualizzato un menu che chiede all'utente di selezionare il tipo di algoritmo da utilizzare per la pianificazione:

Selezionare il tipo di algoritmo:
0) ESCI
1) EDF
2) HPF

Se l'utente inserisce un valore non valido, il software restituisce un messaggio di errore: **"Error: inserimento INPUT MENU (input validi: 0, 1, 2)".**

3.5 ALGORITMI DI PIANIFICAZIONE

- **EDF (Earliest Deadline First)**: Se l'utente sceglie 1, il software pianifica la produzione ordinando i prodotti in base alla scadenza più vicina. In caso di parità, viene considerata la priorità più alta. L'algoritmo carica i puntatori relativi alle scadenze sulla pila e li ordina utilizzando l'algoritmo di ordinamento Bubble Sort.
- **HPF (Highest Priority First)**: Se l'utente sceglie 2, il software pianifica la produzione ordinando i prodotti in base alla priorità più alta. L'algoritmo carica i puntatori relativi alle priorità sulla pila e li ordina utilizzando il Bubble Sort.

3.6 OUTPUT DELLA PIANIFICAZIONE

Dopo l'ordinamento, il software stampa la stringa **"Pianificazione EDF" o "Pianificazione HPF"**, seguita dall'elenco dei prodotti in ordine di produzione con indicato lo slot temporale in cui ciascun prodotto viene messo in produzione.

L'output include anche il tempo di conclusione della produzione e la penalità che l'utente dovrà pagare per gli ordini completati oltre la scadenza definita.

3.7 RIPETIZIONE DEL PROCESSO

Una volta completato l'output, il menu viene nuovamente visualizzato per consentire all'utente di avviare una nuova produzione, magari utilizzando l'algoritmo non scelto precedentemente.

ESEMPIO STACK

4.1 DESCRIZIONE ESEMPIO

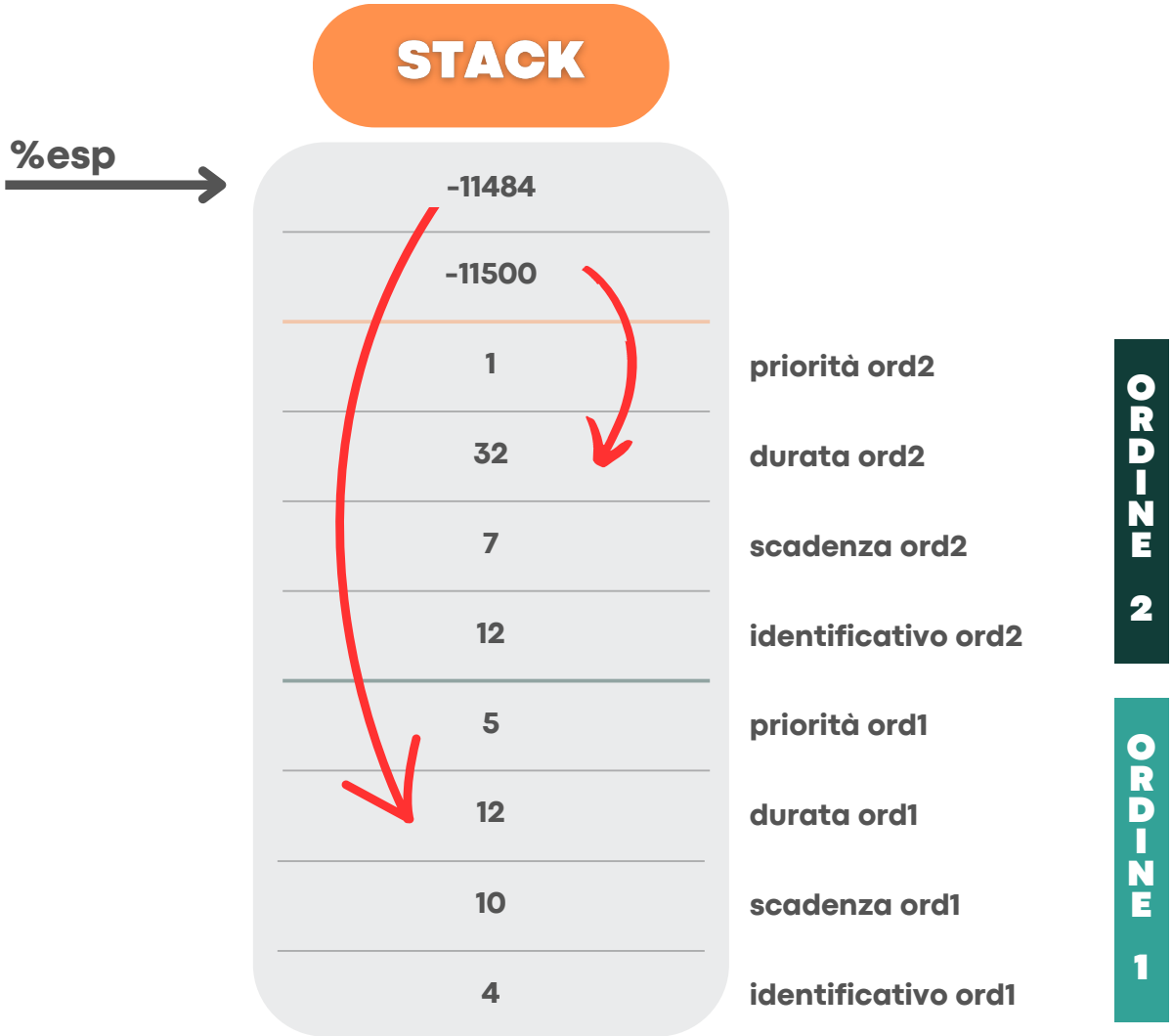
Se nel file .txt troviamo due ordini come segue

Ordini.txt

4,10,12,5

12,7,32,1

Quando il software carica gli ordini in pila e l'utente sceglie l'algoritmo EDF, otterremo la seguente struttura in pila con i parametri relativi agli ordini e i puntatori relativi alle scadenze. (immagine riportata qui sotto)



SCELTE PROGETTUALI

5.1 DESCRIZIONE

1. **Gestione di Più di 10 Ordini:** Il software è progettato per gestire anche più di 10 ordini, tuttavia esiste il rischio che lo stack raggiunga la fine dello spazio disponibile in tali casi.
2. **Utilizzo di Stack e Puntatori:** La scelta di utilizzare uno stack e puntatori è stata effettuata considerando che l'utente inserirà un numero limitato di ordini (massimo 10), quindi i dati da gestire non saranno eccessivi.
3. **Gestione degli Errori**: Il software è progettato per restituire specifici messaggi di errore per aiutare l'utente a capire il problema:
   - "Errore nei valori inseriti (range non rispettati)": Viene mostrato se i valori letti dal file degli ordini non rientrano nei range previsti (identificativo: 1-127, durata: 1-10, scadenza: 1-100, priorità: 1-5).
   - "Errore nell'inserimento dell'input per scegliere l'algoritmo da utilizzare": Viene mostrato se l'utente inserisce un valore non valido durante la scelta dell'algoritmo (validi: 0 per uscire, 1 per EDF, 2 per HPF).
4. **Assunzioni Sintattiche del File degli Ordini**: Si presume che il file degli ordini sia sintatticamente corretto e non contenga spazi tra le virgole o tra le righe.
5. **Condizione per Uscire dal Programma**: Per terminare il programma, l'utente deve inserire il valore 0 durante la selezione dell'algoritmo, come indicato nel menu di scelta.

TEST EFFETTUATI

6.1 DESCRIZIONE TEST

Ipotizzando di avere il file .txt contente i seguenti ordini:

12,7,10,3
7,5,25,1
78,6,32,4
52,9,14,2
2,10,50,5
63,5,46,2
12,9,44,5

Utilizzando l'algoritmo EDF otteniamo il seguente output:

Pianificazione EDF:
12:0
52:7
7:16
78:21
12:27
63:36
2:41
Conclusione: 51
Penalty: 9

Utilizzando l'algoritmo HPF otteniamo il seguente output:

Pianificazione HPF:
12:0
2:9
78:19

12:25
52:32
63:41
7:46
Conclusione: 51
Penalty: 146