|  |
| --- |
| RELAZIONE DI LABORATORIO N.1 |

|  |
| --- |
| Titolo: calcolo dell’espressione: W=X-2Y+3Z, dove i valori sono assegnati in decimale e tra parentesi quelli in esadecimale. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma assembly dovrà svolgere l’espressioen W=X-2Y+3Z. L’unico registro usato sarà l’accumulatore AX. Le moltiplicazioni verranno svolte tramite somme successive: ossia 2Y=Y+Y. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni  Il codice provato funziona e le operazioni di base (moltiplicazione) è stata realizzata tramite l’addizione. |

|  |
| --- |
| RELAZIONE DI LABORATORIO N.2 |

|  |
| --- |
| Titolo: calcolo dell’espressione: W=2X+3Y-2Z, dove i valori sono assegnati in decimale e tra parentesi quelli in esadecimale. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma assembly dovrà svolgere l’espressione W=2X+3Y-2Z. L’unico registro usato sarà l’accumulatore AX. Le moltiplicazioni verranno svolte tramite somme successive: ossia 3Y=Y+Y+Y. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni  Il codice provato funziona e le operazioni di base (moltiplicazione) è stata realizzata tramite l’addizione. |

|  |
| --- |
| RELAZIONE DI LABORATORIO N.3 |

|  |
| --- |
| Titolo: Moltiplicazione per 2 tramite ROL (ROTATE LEFT) del contenuto di un registro. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Tramite l’istruzione ROL (ROTATE LEFT) sarà effettuata la moltiplicazione del contenuto di un registro (AX in questo caso che assume il valore della DW word A1 -> 0002h) per 2, dando come risultato il doppio del suo valore, il risultato sarà poi copiato in un secondo registro di nome RIS e sarà eseguita l’istruzione di return al DOS alla fine. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni  Il codice provato funziona e le operazioni di base (moltiplicazione) è stata realizzata tramite shift del valo. |

|  |
| --- |
| RELAZIONE DI LABORATORIO N.4 |

|  |
| --- |
| Titolo: Moltiplicazione per 8 tramite ROL (ROTATE LEFT) del contenuto di un registro. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Tramite l’istruzione ROL (ROTATE LEFT) sarà effettuata la moltiplicazione del contenuto di un registro (AX in questo caso che assume il valore della DW word A1 -> 0002h) per 8, dando come risultato il la moltiplicazione per 8 del valore, il risultato sarà poi copiato in un secondo registro di nome RIS e sarà eseguita l’istruzione di return al DOS alla fine. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni  Il codice provato funziona e le operazioni di base (moltiplicazione) è stata realizzata tramite shift del valore. |

|  |
| --- |
| RELAZIONE DI LABORATORIO N.5 |

|  |
| --- |
| Titolo: Divisione per 2 con ROR (ROTATE RIGHT) del contenuto di un registro. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Tramite l’istruzione ROR è stato eseguito lo shift di tutti i valori nel registro AX (che ha preso il valore della WORD AX ossia 00A0h) e quindi avviene la divisione del valore per 2 (perché lo shift è stato effettuato solamente una volta.  Testo della funzione scritto nel codice d’esempio:  funzione dell'istruzione ROR (Rotate right).  sposta verso destra tutti i valori e sposta il meno significativo  a destra dove (se presente) c'era quello + significativo.  Quindi sarebbe una sorta di divisione qual ora non sia presente  resto, nel caso sia presente invece ne modifica il valore  che sarà incrementato o decrementato in base ai valori  presenti nel registro, con questa istruzione è possibile  spostare di + di una sola volta tutti i valori (essenzialmente  funziona simile alla ROL solamente che al posto di L c’è R che  sta per right e va verso destra invece di sinistra) specificandolo  in modo simile alla ROL.  Recap di quello detto, è l'opposto del ROL. |

ESEMPIO SOTTO!

V

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati (1 e 2) |

|  |
| --- |
| Conclusioni  Il codice provato ha dato il risultato previsto, 8/2 -> 4/4 -> 1 quindi conferma che funziona. |

|  |
| --- |
| RELAZIONE-Esercizi in classe 1 |

|  |
| --- |
| Titolo: RELAZIONE-Esercizi in classe 1  Scrivere un programma Assembly che visualizzi la scritta HELLO su finestra windows (ho aggiunto world). |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma scriverà ogni singolo carattere passo a passo in un nuovo DOS, con AH 2 comunico  Le modalità di output e poi assegno al registro DL ogni singolo carattere e lo scrivo in uscita al monitor  Con INT 21H |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Interazione con monitor avvenuta con successo, la scritta è stata scritta come previsto. |

|  |
| --- |
| Esercitazione 3 |

|  |
| --- |
| Titolo: Esercitazione 3.1  Modificare programma precedente di scrittura Hello e aggiungere in una nuova linea la scritta  IT IS A GOOD DAY. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Usando l’esercizio della scorsa volta, aggiungo il testo nuovo ma prima anche il valore 0DH al registro DL, poi lo scrivo in output con INT 21H.  Creo una nuova linea, assegnando a DL il valore 0AH e lo scrivo in output, poi per il resto faccio come prima per ogni singolo carattere della nuova parte. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Interazione con monitor avvenuta con successo, la scritta è stata scritta come previsto. |

|  |
| --- |
| Esercitazione 3 Parte 2 |

|  |
| --- |
| Titolo: Esercitazione 3.1  Programma che prende carattere in input e lo ripete. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma dovrà chiedere un input dall’utente per poi scriverne un risultato in output ripetendo il carattere scritto. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Interazione con input tastiera e ripetizione carattere in input salvato a registro. |

|  |
| --- |
| Esercitazione 4 |

|  |
| --- |
| Titolo: Esercitazione 4  Programma che scrive messaggio di saluto sul video, tramite una stringa e servizio DOS. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma utilizzerà un servizio del DOS per scrivere una stringa a video. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Scrittura a video tramite servizio del DOS di una stringa, il risultato ottenuto è stato quello previsto. |

|  |
| --- |
| Esercitazione 13 |

|  |
| --- |
| Titolo: Esercitazione 13  Somma di valori presenti in un vettore (VERSIONE I) |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma assegnerà un vettore con variabili di tipo word e le utilizzerà per eseguirne la somma ad AX. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Somma utilizzando valori WORD presenti in un vettore nel registro AX effettuata con successo. |

|  |
| --- |
| Esercitazione numero **perso** |

|  |
| --- |
| Titolo: Esercitazione **persa** Scrittura di più stringhe e utilizzo 13,10 per andare a capo su DOS. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Utilizzando l’istruzione 13,10 si andrà a scrivere delle stringhe a capo tra di loro in DOS. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Scrittura in DOS di stringhe/word andando a capo con l’istruzione 13,10 |

|  |
| --- |
| Esercitazione 14 |

|  |
| --- |
| Titolo: Esercitazione 14 Somma di valori presenti in un vettore (VERSIONE II) e ciclo. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma assegnerà un vettore con variabili di tipo word e le utilizzerà per eseguirne la somma ad AX senza forzarle questa volta + ciclo. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Somma utilizzando valori WORD presenti in un vettore nel registro AX effettuata con successo e ciclo. |

|  |
| --- |
| Esercitazione 15 |

|  |
| --- |
| Titolo: Esercitazione 15 Somma di valori presenti in un vettore (VERSIONE III) e loop. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma assegnerà un vettore con variabili di tipo word e le utilizzerà per eseguirne la somma ad AX senza forzarle questa volta + loop per il numero di volte specificato. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Somma utilizzando valori WORD presenti in un vettore nel registro AX effettuata con successo e utilizzo del LOOP. |

|  |
| --- |
| Esercitazione 16 |

|  |
| --- |
| Titolo: Esercitazione 16 Somma di valori presenti in un vettore (VERSIONE III) e loop. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma assegnerà un vettore con variabili di tipo word e le utilizzerà per eseguirne la somma ad AX senza forzarle questa volta + loop per il numero di volte specificato. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Somma utilizzando valori WORD presenti in un vettore nel registro AX effettuata con successo e utilizzo del LOOP. |

|  |
| --- |
| Esercitazione 17 |

|  |
| --- |
| Titolo: Esercitazione 17 Somma di valori presenti in un vettore (VERSIONE III) e loop. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma assegnerà un vettore con variabili di tipo word e le utilizzerà per eseguirne la somma ad AX senza forzarle questa volta + loop per il numero di volte specificato. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Somma utilizzando valori WORD presenti in un vettore nel registro AX effettuata con successo e utilizzo del LOOP. |

|  |
| --- |
| Esercitazione 18 |

|  |
| --- |
| Titolo: Esercitazione 18  Somma di valori presenti in un vettore (VERSIONE IV) |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma assegnerà un vettore con variabili di tipo word e le utilizzerà per eseguirne la somma ad AX. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Somma utilizzando valori WORD presenti in un vettore nel registro AX effettuata con successo. |

|  |
| --- |
| Esercitazione 19 |

|  |
| --- |
| Titolo: Esercitazione 19  Somma di valori presenti in un vettore (VERSIONE V) |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma assegnerà un vettore con variabili di tipo word e le utilizzerà per eseguirne la somma ad AX. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Somma utilizzando valori WORD presenti in un vettore nel registro AX effettuata con successo. |

|  |
| --- |
| Esercitazione 20 |

|  |
| --- |
| Titolo: Esercitazione 20 Somma di valori presenti in un vettore (VERSIONE VI) e jump. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma assegnerà un vettore con variabili di tipo word e le utilizzerà per eseguirne la somma ad AX senza forzarle questa volta + loop per il numero di volte specificato. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Somma utilizzando valori WORD presenti in un vettore nel registro AX effettuata con successo e utilizzo del LOOP. |

|  |
| --- |
| Esercitazione 21 |

|  |
| --- |
| Titolo: Esercitazione 21 Somma di valori presenti in un vettore (VERSIONE VII) e jump. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma assegnerà un vettore con variabili di tipo word e le utilizzerà per eseguirne la somma ad AX senza forzarle questa volta + loop per il numero di volte specificato. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Somma utilizzando valori WORD presenti in un vettore nel registro AX effettuata con successo e utilizzo del LOOP. |

|  |
| --- |
| Esercitazione 22 |

|  |
| --- |
| Titolo: Esercitazione 22 Somma di valori presenti in un vettore (VERSIONE VIII) e loop. |

|  |
| --- |
| Descrizione di ciò che dovrà svolgere il programma Assembly:  Il programma assegnerà un vettore con variabili di tipo word e le utilizzerà per eseguirne la somma ad AX senza forzarle questa volta + loop per il numero di volte specificato. |

|  |
| --- |
| Screenshot codifica in Assembly |

|  |
| --- |
| Screenshot risultati |

|  |
| --- |
| Conclusioni:  Somma utilizzando valori WORD presenti in un vettore nel registro AX effettuata con successo e utilizzo del LOOP. |