[Sessione 1] Operazioni aritmetiche, accesso alla memoria

[s1.1] Store and sum

nome del file sorgente: storesum.asm

Si scriva il codice Assembly che:

- carichi il valore 5 nel registro $s1;
- carichi il valore 7 nel registro $s2;
- carichi il valore della somma dei due nel registro $s0.

[s1.2] Calcolo di un espressione

nome del file sorgente: expression.asm

Si traduca in Assembly la seguente riga di codice:

```
A = B+C-(D+E)
```

assegnando alle variabili

```
A, B, C, D, E
```

i registri

```
$s0, ..., $s4.
```

Si assumano valori iniziali

```
1, 2, 3, 4
```

[s1.3] Moltiplicazione e divisione

nome del file sorgente: muldiv.asm

Si implementi il codice Assembly che effettui la moltiplicazione e la divisione tra i numeri 100 e 45, utilizzando le istruzioni dell'ISA e le pseudoistruzioni.

[s1.4] Allocazione di un array nel segmento dati

nome del file sorgente: array4.asm

Mediante le direttive assembler, si allochi la memoria per un array di dimensione 4 inizializzato in memoria come segue:

```
A[0]=0,
A[1]=4,
A[2]=8,
A[3]=12
```

[s1.5] Lettura da memoria e scrittura in array

nome del file sorgente: arraysum.asm

Si scriva il codice Assembly che effettui:

```
A[12] = h + A[8];
```

Si assuma che:

- h sia una variabile memorizzata nel segmento dati;
- A sia un array di 15 elementi memorizzato nel segmento dati;

Si inizializzino h e A con valori a piacere utilizzando delle direttive per l'assembler nel segmento dati.

[s1.6] Lettura e scrittura da memoria (1)

nome del file sorgente: rwmemoria.asm

Si scriva il codice Assembly che effettui:

```
A[99] = 5 + B[i] + C
```

Si assuma che:

- A e B siano vettori di 100 elementi, ogni elemento è un intero a 32 bit;
- C e i siano variabili intere a 32 bit.

Si inizializzino unicamente questi dati:

```
i=3, C=2, B[i]=10.
```

[s1.7] Lettura e scrittura da memoria (2)

nome del file sorgente: rwmemoria2.asm

Si scriva il codice Assembly che effettui:

```
A[c-1] = c*(B[A[c]] + c)/A[2*c-1]
```

Si assuma che:

- A e B siano vettori di elementi, ogni elemento è un intero a 32 bit;

- c sia una variabile intera a 32 bit.

Si inizializzino i dati in memoria in questo modo:

```
c=2
A[0]=-1
A[1]=-1
A[2]= 1
A[3]= 4
B[0]=-1
B[1]= 6
B[2]=-1
B[3]=-1
```