1. [10] Si traduca in linguaggio Assembly MIPS nativo (evitando di utilizzare pseudoistruzioni) la seguente procedura in linguaggio C, che calcola la potenza di due elevato all'esponente fornito come argomento.

La procedura si aspetta l'argomento `n` nel registro `$a0` e restituisce il risultato in `$v0`. Si supponga che gli argomenti siano sempre numeri interi non negativi e minori di 32.

```
int PowerOfTwo( int n )
{
     if( n < 1 )
          return( 1 );
     else
          return( 2 * PowerOfTwo(n-1) );
}
```

2. [10] Si scriva un programma Assembly completo, per ambiente SPIM, che permetta ad un utente di calcolare la potenza di due di un esponente fornito dall'utente. Il programma esegue il calcolo chiamando la funzione `PowerOfTwo` descritta nell'esercizio precedente. Se l'esponente fornito in ingresso è negativo o non minore di 32, il programma ripete la richiesta di esponente in ingresso. Nel caso invece di esponente valido, una volta effettuato il calcolo, il programma termina.

Il programma deve presentarsi a terminale come nel seguente esempio:

```
Inserire il valore dell'esponente
> 43
Esponente non valido!
Inserire il valore dell'esponente
> 5
Due elevato a 5 risulta: 32
```

3. [6] Si traduca il seguente frammento di codice: a) in Assembly MIPS nativo e b) in linguaggio macchina, specificando valore e ampiezza in bit dei campi di ogni istruzione:

```
test: li $a0, 8^8
      divi $s1, $t1, 12    # divide by immediate
      bgei $s1, -6, test   # branch if greater or equal than
```

5. [4] Rappresentare gli indirizzi ed il contenuto, byte per byte (in formato esadecimale), del segmento dati della memoria, a seguito dell'esecuzione delle seguenti direttive:

```
.data 0x300
.word 1024, -8
.byte 24, -24
.half 0x10
```


System calls

	codice ($v0)	argomenti	risultato
print_int	1	$a0	
print_float	2	$f12	
print_double	3	$f12	
print_string	4	$a0	
read_int	5		$v0
read_float	6		$f0
read_double	7		$f0
read_string	8	$a0,$a1	
sbrk	9	$a0	$v0
exit	10		

Registri MIPS

0	zero		24-25	t8 – t9
1	at		26-27	k0 – k1
2-3	v0 – v1		28	gp
4-7	a0 – a3		29	sp
8-15	t0 – t7		30	s8
16-23	s0 – s7		31	ra

MIPS Instruction Set: