CS2100 Tutorial 4

AY 24/25 Sem 2 — github/omgeta

Q1.

	Operand	Target Memory Address	Content
(a)	$t1	Not applicable	15000
	$t2	Not applicable	20000
(b)	$s2	Not applicable	200
	100($zero)	100	1000
(c)	$t4	Not applicable	30000
	40($s2)	240	2400
(d)	$s3	Not applicable	240
	200($zero)	200	2000
(e)	$t3	Not applicable	25000
	$zero($t1)	15000	150
(f)	$s1	Not applicable	160
	$140($s1)	300	3000

Q2. (a.) Stack:

```
push  @a1
push  @a2
add
pop   @a0
push  @a0
push  @a2
add
pop   @a1
push  @a0
push  @a1
add
pop   @a2
```

(b.) Accumulator:

```
load  @a1
add   @a2
store @a0
add   @a2
store @a1
add   @a0
store @a2
```

(c.) Memory-Memory:

```
add @a0, @a1, @a2
add @a1, @a0, @a2
add @a2, @a0, @a1
```

(d.) Register-Register:

```
load  $r1, @a1
load  $r2, @a2
add   $r0, $r1, $r2
store $r0, @a0
add   $r1, $r0, $r2
store $r1, @a1
add   $r2, $r0, $r1
store $r2, @a2
```

Q3. (a.)

	Number of bits for longest instructions	Number of bytes
Stack	10	2
Accumulator	10	2
Memory-Memory	24	4
Register-Register	13	2

(b.) Stack: $12 \times 2 = 24$ bytes
Accumulator: $7 \times 2 = 14$ bytes
Memory-Memory: $3 \times 4 = 12$ bytes
Register-Register: $8 \times 2 = 16$ bytes
Therefore, Memory-Memory is the most efficient in terms of code-size

Q4. (a.) To find minimum, maximise opcode bits of B (smaller opcode space):
$A + B = (1 \cdot 2^5) + (2^6 - 1) = 95$

(b.) To find maximum, minimise opcode bits of B (smaller opcode space):
$A + B = ((2^6 - 1) \cdot 2^5) + (1) = 2017$