CPSC 213, Winter 2016, Term 1

Midterm II Sample Questions - Solutions

Exercise 1

```
doit:
      deca r5           # allocate space for ra on stack
      st r6, (r5)       # save ra on stack
      deca r5           # make room for argument on stack
      ld $5, r0         # r0 = 5
      st r0, (r5)       # arg0 = 5
      gpc $6, r6        # get return address
      j add             # call addOne (5)
      inca r5           # remove argument area
      ld $x, r1         # r1 = &x
      st r0, (r1)       # x = addOne (5)
      ld (r5), r6       # restore ra from stack
      inca r5           # remove ra space from stack
      j (r6)            # return
addOne:
      ld (r5), r0       # r0 = a
      inc r0            # r0 = a + b
      j (r6)            # return a + b
```

Exercise 2

```
      countZero:  ld $len, r1               # r1 = address of len
                  ld (r1), r1               # r1 = len
                  ld $a, r2                 # r2 = address of a
                  ld (r2), r2               # r2 = a
                  ld $0, r0                 # r0 = c

      loop:       bgt r1, cont              # goto cont if len>0
                  br done                   # goto done if len<=0

      cont:       dec r1                    # len = len - 1
                  ld (r2, r1, 4), r3        # r3 = a[len]
                  beq r3, loop              # goto loop if a[len]==0
                  inc r0                    # c=c+1 if a[len]!=0
                  br loop                   # goto loop
      done:       j (r6)                    # return c
```

Exercise 3

```
      ld $0, r0
      ld $s, r1
      ld (r1), r2
      st r0, 24(r2)
```

Exercise 4

It prints 120.

Exercise 5

Yes, a memory leak is possible. It can be fixed as follows:

```
char* copy (char* from, int n) {
  char* to = malloc (n);
  for (int i=0; i<n; i++)
    to[i] = from[i];
  return to;
}
void foo (char* x, int n) {
  char* y = copy (x, n);
  printf ("%s", y);
  free (y);
}
```

Exercise 6

```
void getsum (char* buf, int n) {
  int s=0;
  for (int i=0; i<256; i++)
    s += buf[i];
  printf ("%d\n", s);
}

char buf[256];

void ps() {
  int s = async_read (1234, buf, 256, getsum);
}
```