# ARM ISA Exercises

Young W. Lim

June 8, 2016

# Assembly

```
a = b + c;
a = b - c;
a = b + c - d;
a = b + c;
a = b + c - d;
```

# Arithmetic

```
int a = 0x4f3C
int a = 0x6d5e4f3c
```

# Conditional Statement

```
if (i == j) f = g + h;
f = f - 1;

if (i == j) f = g + h;
else f = f - 1;

switch (amount) {
  case 20: fee = 2; break;
  case 50: fee = 3; break
  case100: fee = 5; break;
  default: fee = 0;
}
```

# Loop (1)

```
int pow = 1;
int x = 0;
while (pow != 128) {
  pow = pow * 2;
  x = x + 1;
}

int sum = 0;
for (i=0; i != 10; i=i+1) {
  sum = sum + i;
}
```

# Loop (2)

```
int sum = 0;
int i = 0;
while (i != 10) {
  sum = sum + i;
  i = i + 1;
}
```

```
int sum = 0;
for (i=1; i<101; i=i*2) {
  sum = sum + i;
}
```

# Arrays

```
int array[5];
array[0] = array[0] * 8;
array[1] = array[1] * 8;

int i;
int array[1000];
for (i=0; i<1000; i=i+1) {
  array[i] = array[i] * 8;
}
```

# Procedure Calls (1)

```
int main() {
  simple();
  ...
}

void simple() {
  return;
}
```

# Procedure Calls (2)

```
int main() {
  int y;
  ...
  y = diffofsums(2, 3, 4, 5);
  ...
}

int diffofsums(int f, int g, int h, int i) {
  int result;
  result = (f+g) - (h+i);
  return result;
}
```

# Recursive Calls

```
int factorial (int n) {
  if (n <= 1)
    return 1;
  else
    return (n * factorial(n-1));
}
```

# Object Code and Executable

```
int f, g, y;

int main (void) {
  f = 2;
  g = 3;
  y = sum(f, g);
  return y;
}
```

# Reference

[1] D. Harris, Digital Design and Computer Architecture", 2nd ed.
[2] D.A. Patterson & J.H. Hennessy, Computer Organization and Design (ARM ed)