# Princeton University
# COS 217: Introduction to Programming Systems
# A Subset of IA-32 Assembly Language for the Assembler Assignment

Your assembler should handle these IA-32 instructions:

## Directives

```
.section ".sectionname"
.skip n
.align n
.byte bytevalue1, bytevalue2, ...
.long longvalue1, longvalue2, ...
.ascii "string1", "string2", ...
.asciz "string1", "string2", ...
.globl label1, label2, ...
```

## Mnemonics

```
Note:  Registers are encoded as three-bit numbers:

EAX=000, EBX=011, ECX=001, EDX=010, ESI=110, EDI=111, EBP=101, ESP=100,
AL=000, AH=100, BL=011, BH=111, CL=001, CH=101, DL=010, DH=110
```

### Data Transfer Mnemonics

```
movl $immed, %destreg
10111ddd iiiiiiii iiiiiiii iiiiiiii iiiiiiii
Note: The assembler stores immed in two's complement little-endian form.

movl $label, %destreg
10111ddd 00000000 00000000 00000000 00000000
Note: The assembler generates a relocation record for the last four bytes.

movl %sourcereg, %destreg
10001001 11sssddd

movl %sourcereg, number(%destreg)
10001001 10sssddd nnnnnnnn nnnnnnnn nnnnnnnn nnnnnnnn
Note: The assembler stores number in two's complement little-endian form.

movl number(%sourcereg), %destreg
10001011 10dddsss nnnnnnnn nnnnnnnn nnnnnnnn nnnnnnnn
Note: The assembler stores number in two's complement little-endian form.

movb $immed, %destreg
10110ddd iiiiiiii

movb %sourcereg, %destreg
10001000 11sssddd

movb %sourcereg, number(%destreg)
10001000 10sssddd nnnnnnnn nnnnnnnn nnnnnnnn nnnnnnnn
Note: The assembler stores number in two's complement little-endian form.

movb number(%sourcereg), %destreg
10001010 10dddsss nnnnnnnn nnnnnnnn nnnnnnnn nnnnnnnn
Note: The assembler stores number in two's complement little-endian form.
```

**pushl %*sourcereg***
01010sss

**popl %*sourcereg***
01011sss

## Arithmetic Mnemonics

**addl %*sourcereg*, %*destreg***
00000001 11sssddd

**subl %*sourcereg*, %*destreg***
00101001 11sssddd

**imull %*sourcereg***
11110111 11101sss

**idivl %*sourcereg***
11110111 11111sss

## Control Transfer Mnemonics

**cmpl %*sourcereg*, %*destreg***
00111001 11sssddd

Note: For all jump and call instructions, the last four bytes are a positive or negative byte displacement in two's complement little-endian form. The default displacement is -4. The assembler generates a relocation record for those last four bytes when it must.

**jmp *label***
11101001 11111100 11111111 11111111 11111111

**je *label***
00001111 10000100 11111100 11111111 11111111 11111111

**jne *label***
00001111 10000101 11111100 11111111 11111111 11111111

**jl *label***
00001111 10001100 11111100 11111111 11111111 11111111

**jle *label***
00001111 10001110 11111100 11111111 11111111 11111111

**jg *label***
00001111 10001111 11111100 11111111 11111111 11111111

**jge *label***
00001111 10001101 11111100 11111111 11111111 11111111

**call *label***
11101000 11111100 11111111 11111111 11111111

**ret**
11000011