Codes-
Switch statement
for loop
addition
Lab 8 similar structure

RISC/CISC -choose options
CISC-

1. Simple software, complex hardware
2. Most operands can access memory
3. Low number of registers
4. Instructions can have multiple clock cycles
5. Encoded instructions vary in size

RISC-

1. Simple hardware, complex software
2. Only Load/Store instructions can access memory
3. Higher number of registers
4. Instructions tend towards one per clock cycle
5. Encoded instructions are all the same size.

Use of ret and call
Return-
this marks the end of subroutine
Call-
this transfer control to subroutine

Contents of stack frame in order

1. Input Parameter
2. Caller Return Address
3. Caller Base Pointer
4. Local Variables
5. Saved Registers

What are status registers and what do they hold?
Status registers are special registers that hold the Boolean information about the processor's current state

Old vocabulary-

1. When register holds an address and it is used to access memory
   indirect
2. The main, hidden part of the operating system
   kernel
3. When an application asks the operating system to perform a task for it, it uses this unique value.
   System call number
4. Normal applications run in this processor mode
   user

5. Set by the comparison instruction and the used by conditional jump statement
   flags
6. Operating system run in this processor mode
   privileged
7. This defines the order of parameters in a stack frame
   calling convention
8. When hardware needs to contact the operating system, this signal is sent
   interrupt
9. This is a table of addresses, stored on the processor, that reacts when the processor is alerted
   vector
10. When data is read/writer past the end of buffer
    overflow
11. This term is used to refer to all the registers on the processor
    file
12. These registers don't have a specific use and are available to your program.
    general purpose
13. Programs are a combination of these - which are often created by different developers
    object
14. Java (and other high-level programming languages) can be converted into assembly using this
    compiler
15. In assembly, these tell the assembler to allocate space, start a section, etc...
    directive
16. Assembly uses these easy to remember names to identify instructions
    mnemonic
17. This is the first-generation programming language
    machine language
18. The tab and new line characters are classified as this
    control
19. In assembly, this term means the actual raw value
    immediate
20. Each instruction has a unique identifying sequence of
    opcode

Von Neuman architecture attributes

1. Programs are stored and executed in memory
2. Separation of processing from memory
3. Different system components communicate over a shared bus

3 components of bus

1. Address bus- used by processor to access a specific piece of data
2. Data bus- actual data travels over this
3. Control bus- it controls the timing and synchronizes the subsystems.

Signed magnitude
1's complement
Value of registers

Extend 2's complement-
Copy the most significant bit (first bit) and add 8 of them to start

Extend signed-magnitude-
Add 0's after the first bit

Extend 1's complement-
Copy the most significant bit (first bit) and add 8 of them to start

1's complement-
Convert all 1's to 0's and vice versa.

2's complement-
Do 1's complement and add 1.

Calculate decimal value of signed magnitude-
use the most significant bit to write the sign and then calculate the decimal value normally without the most significant bit. The range goes from -127 to 127.

64-Bit registers
RAX, RBX, RCX, RDX, RSI, RDI, RBP, RSP

32-Bit registers
EAX, EBX, ECX, EDX, ESI, EDI, EBP, ESP

16-Bit Registers
AX, BX, CX, DX, BP, SP, SI, DI

8-Bit High
AH, BH, CH, DH

8-Bit Low
AL, BL, CL, DL, SIL, DIL BPL, SPL

New 64-Bit registers
R8, R9, R10, R11, R12, R13, R14, R15

New 32-Bit registers
R8d, R9d, R10d, R11d, R12d, R13d, R14d, R15d

New 16-Bit registers
R8w, R9w, R10w, R11w, R12w, R13w, R14w, R15w

New 8-Bit registers
R8b, R9b, R10b, R11b, R12b, R13b, R14b, R15b

Multiplication
1st operand is RAX
2nd operand must be another register/ memory location
RAX has lower 8 bytes, RDX has upper 8 bytes of result.

Division
IDIV should always be writer after CQO
Numerator→ RDX has upper 8 bytes, RAX has lower 8 bytes.
Denominator→ always another register.

Quotient→ stored in RAX
Remainder→ stored in RDX

Space allocation-
.ascii→ length of string
.quad→ 8 bytes
.byte→ 1byte
.space→ of the size mentioned first

Meaning of each item in a UNIX command
as→ the assembling of the program
ld→ links all the object files
-o→ the next file is output file
lab2.o→ the output file is made
lab2.asm→ the assembled program file.