(5) Question 1. \(1120 = 1*16^3+1*16^2+2*16^1 = 4096+1*256+2*16 = 4096+256+32 = 4384\)

(6) Question 2.
Part a) Call graph
Part b) Precision
Part c) I/O mapped I/O

(6) Question 3. Consider \(\text{ldab } #100 \ \text{subb } #-90\)
To calculate the overflow (V) bit, convert to signed, and operate. \(100 - -90 = 190\) is incorrect, so \(V = 1\)
To calculate the carry (C) bit, convert to unsigned, and operate. \(100 - 166 = -66\) is incorrect, so \(C = 1\)

(5) Question 4. \(2^{10}\) is about \(10^3\), so \(2^{20}\) is about \(10^6\), which is 6 decimal digits.

(10) Question 5. Interface the switch to PT0 using negative logic

(5) Question 6. Current is exponentially related to voltage

(5) Question 7. \$6000 D001 subb $0001

<table>
<thead>
<tr>
<th>R/W</th>
<th>Addr</th>
<th>Data</th>
<th>Changes to A, B, X, Y, S, PC, IR, EAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>$6000</td>
<td>$D0</td>
<td>IR=$D0, PC=$6001</td>
</tr>
<tr>
<td>R</td>
<td>$6001</td>
<td>$01</td>
<td>EAR=$0001, PC=$6002</td>
</tr>
<tr>
<td>R</td>
<td>$0001</td>
<td>$03</td>
<td>B = $45-$03 = $42</td>
</tr>
</tbody>
</table>

(4) Question 8. Subroutine return address is the address of the instruction after the \textbf{bsr}. Push $411B when \textbf{Test} called, push $4115 when \textbf{Delay} called. 16-bit numbers are stored big endian.
(4) **Question 9.** C code to create a variable named `Position` with range 0 to 65535

```c
unsigned short Position;
```

(10) **Question 10.** You only have to push one of them

```assembly
pshd
  tfr Y,D ; Y goes to D
  tfr X,Y ; X goes to Y
  pulx ; D goes to X
```

It works, but is less efficient to push all three

```assembly
pshd ; D on stack
pshx ; X,D (X on top)
pshy ; Y,X,D (Y on top)
puld ; Y goes to D
puly ; X goes to Y
pulx ; D goes to X
```

(20) **Question 11.** Two positive logic switches are connected to PT5 and PT2, and one positive logic LED is connected to PT0. Turns on the LED if exactly one of the two switches is on.

```assembly
org  $4000
main  lds #$4000 ;optional here because stack not used
  bset DDRT,#$01 ;PT0 output
  bclr DDRT,#$24 ;PT5 and PT2 inputs
loop  ldaa PTT
  anda #$24 ;RegA is 00,04,20 or 24
  cmpa #$20 ;PT5 set, PT2 clear
  beq  On
  cmpa #$04 ;PT2 set, PT5 clear
  beq  On
Off  bclr PTT,#$01 ;LED off
   bra  loop
On   bset PTT,#$01 ;LED on
   bra  loop
org  $FFFE
fdb  main ;reset vector
```

A second possible solution without conditionals, PT0 = PT5^PT2

```assembly
org  $4000
main  lds #$4000 ;optional here because stack not used
  ldaa DDRT
  oraa #$01 ;PT0 output
  anda %11011011 ;PT5 and PT2 inputs
  staa DDRT
loop  ldaa PTT ;read Bit 5
  lsra
  lsra
  lsra ;bit2 has PT5 value
  eora PTT ;bit2 has PT5^PT2
```
lsra
lsra ;bit0 has PT5^PT2
anda #1 ;RegA has PT5^PT2
ldab PTT
andb #$FE ;RegB has original PT7-PT1
aba ;combine
staa PTT ;PT0 = PT5^PT2
bra loop
org $FFFE
fdb main ;reset vector

A third possible solution with fewest instructions
org $4000
main lds #$4000 ;optional here because stack not used
bset DDRT,#$01 ;PT0 output
bclr DDRT,#$24 ;PT5 and PT2 inputs
loop brset PTT,#$20,Is5 ;go to is5 if PT5 is high
no5 brset PTT,#$04,On ;turn on if PT5=low and PT2=high
Off bclr PTT,#$01 ;LED off
bra loop
is5 brset PTT,#$04,Off ;turn off if PT5=high and PT2=high
On bset PTT,#$01 ;LED on
bra loop
org $FFFE
fdb main ;reset vector

(20) Question 12. Write a C program that controls a kidney dialysis pump.
void main(void){
    DDRT = 0xFF; // output power to pump
    DDRP = 0x00; // input flow rate in ml/min
    while(1){
        if(PTP < 100){ // too slow
            if(PTT < 255){
                PTT++; // increase power
            }
        }
        if(PTP > 100){ // too fast
            if(PTT > 0){
                PTT--; // decrease power
            }
        }
    }
}

Just for fun, open assemble and run the motor.uc example in TExaS.