Homework Set 2 for Module 1

Due at the Beginning of Class (12:30 pm) on Friday, September 14

Name: ____ **Class No: __ __ __ __ - __**

Signature: _______________________________________ **Score: _______ / 100**

1. [15 points] Compile the following C program fragment using Code Warrior and "step" through its execution. Then, compile this C code fragment "by hand" into HC(S)12 assembly code; assemble and run your hand-compiled code to verify it works as expected.

```
i=3;

if(i==1)
   pay=100;
else if(i==2)
   pay=200;
else if(i==3)
   pay=300;
else if(i==4)
   pay=400;
```

2. [15 points] Compile the following C program fragment using Code Warrior and "step" through its execution. Then, compile this C code fragment "by hand" into HC(S)12 assembly code; assemble and run your hand-compiled code to verify it works as expected.

```
for(i=1;i<5;i++) {

if(i==1)
   pay=100;
else if(i==2)
   pay=200;
else if(i==3)
   pay=300;
else if(i==4)
   pay=400;

}
```

3. [15 points] Compile the following C program fragment using Code Warrior and "step" through its execution. Then, compile this C code fragment "by hand" into HC(S)12 assembly code; assemble and run your hand-compiled code to verify it works as expected.

```
unsigned int i;
unsigned long pay;

i=3;

switch(i) {
     case 1:
          pay=100;
          break;
     case 2:
          pay=200;
          break;
     case 3:
          pay=300;
          break;
     case 4:
          pay=400;
          break;
     default:
          pay=0;
}
```

4. [5 points] Compile the following C program fragment using Code Warrior and “step” through its execution. Write down the sequence of numbers generated for the variable “`j`” and explain what the sequence represents.

```
unsigned int i;
unsigned int j;
unsigned int n;

n=5;
j=0;

for(i=1;i<=10;i++) {

j=(j+1)%n;

}
```

5. [5 points] Compile the following C program fragment using Code Warrior and "step" through its execution. Write down the sequence of numbers generated for the variable "j" and explain what the sequence represents.

```
unsigned int i;
unsigned int j;
unsigned int n;

i=0;
n=5;

while(i<10) {

j=(i++)%n;

}
```

6. [5 points] Compile the following C program fragment using Code Warrior and “step” through its execution. Write down the sequence of numbers generated for the variable “j” and explain what the sequence represents.

```
unsigned int i;
unsigned int j;
unsigned int n;

i=0;
n=5;

do {

j=(i++)%n;

}  while(i<10);
```

7. [10 points] Write (and test) a macro `imodb` that increments the contents of a *byte* memory location a given (unsigned integer) modulus, where the first substitution parameter is the memory location and the second substitution parameter is the modulus. Write your macro such that it does not use any labels, i.e., use the symbol for the location counter ($) instead.

   Examples:

   ```
   imodb    memb,10     ;(memb) ← [(memb)+1] % 10
   imodb    memb,100    ;(memb) ← [(memb)+1] % 100
   ```

   where

   ```
   memb   rmb   1
   ```

   Hint: See previous three problems!

8. [10 points] Write (and test) a macro `imodw` that increments the contents of a *word* memory location a given (unsigned integer) modulus, where the first substitution parameter is the memory location and the second substitution parameter is the modulus. Write your macro such that it does not use any labels, i.e., use the symbol for the location counter ($) instead.

   Examples:

```
imodw    memw,100      ;(memw) ← [(memw)+1] % 1000
imodw    memw,10000    ;(memw) ← [(memw)+1] % 10000
```

   where

```
memw   rmb   2
```

9. [20 points] Given the following three loop structures discussed in class:

METHOD 1:

```
bgmac ldx   #XA      (2)
      ldy   #YA      (2)
      movw  #0,MA    (5)
      movw  #0,MA+2  (5)
      clrb           (1)
loop  cmpb  #N       (1)
      beq   exit   (3/1)
      emacs MA      (13)
      leax  2,x      (2)
      leay  2,y      (2)
      incb           (1)
      bra   loop     (3)
exit  rts            (5)

N     equ   ?
XA    fdb   ?,?,?
YA    fdb   ?,?,?
MA    rmb   4
```

METHOD 2:

```
bgmac ldx   #XA        (2)
      ldy   #YA        (2)
      movw  #0,MA      (5)
      movw  #0,MA+2    (5)
      ldab  #N+1       (1)
loop  dbeq  b,exit   (3/3)
      emacs MA        (13)
      leax  2,x        (2)
      leay  2,y        (2)
      bra   loop       (3)
exit  rts              (5)

N     equ   ?
XA    fdb   ?,?,?
YA    fdb   ?,?,?
MA    rmb   4
```

METHOD 3:

```
bgmac ldx   #XA        (2)
      ldy   #YA        (2)
      movw  #0,MA      (5)
      movw  #0,MA+2    (5)
      ldab  #N         (1)
loop  emacs MA        (13)
      leax  2,x        (2)
      leay  2,y        (2)
      dbne  b,loop (3/3)
exit  rts              (5)

N     equ   ?
XA    fdb   ?,?,?
YA    fdb   ?,?,?
MA    rmb   4
```

(a) [8 points] Write a ***formula*** that describes the <u>calculation</u> done by the "bgmac" subroutine.

(b) [12 points] Write a ***formula*** for the *total execution time* (in machine cycles) of each of the subroutines, above, as a function of **N** (for **N** ≥ 1). Note that the cycle count for each instruction is given; for conditional branches, these are given as cycles for the branch <u>taken</u> / <u>not taken</u>.

Method 1	Method 2	Method 3