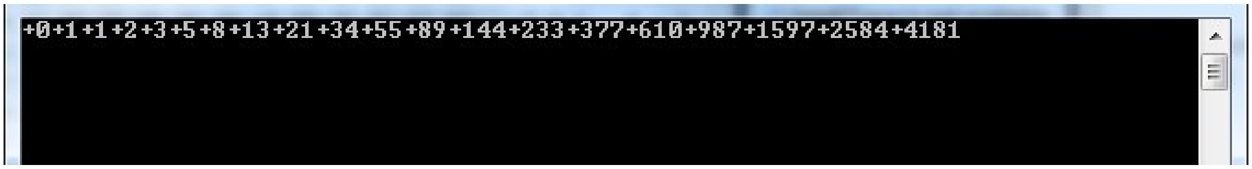
HW 6: THEME: Procedures

1. A Fibonacci number is calculated by summing the preceding two Fibonacci numbers: the fifth number is 3, the summation of 1 and 2. As such, $f_n = f_{n-1} + f_{n-2}$. In this sequence, $f_0 = 0$ and $f_1 = 1$: {0, 1, 1, 2, 3, 5, 8, 13, etc}. Draft a program that calculates the first 20 values of the Fibonacci sequence. Your program must utilize a procedure for the Fibonacci calculation.

```
TITLE Fibbonacci Sequence

INCLUDE Irvine32.inc
.data
         n WORD 20 DUP(0)                               ;create 20 number slots
.code
main PROC

         MOV esi, OFFSET n                              ;set pointer to n
         MOV WORD PTR [esi], 0                          ;set first number

         ADD esi, TYPE n                                ;increment pointer
         MOV WORD PTR [esi], 1                          ;set second number

         MOV ecx, 18                                    ;set loop counter to 18

         call CalNext                                   ;calculate next FBNCI number 'ecx' times

         MOV ecx, 20                                    ;set loop counter to 20
         MOV esi, OFFSET n                              ;set pointer to n

         L2:
                   MOV ax, WORD PTR [esi]               ;store current number in eax
                   call WriteInt                        ;print current value

                   ADD esi, TYPE n                      ;increment esi
         LOOP L2

         exit
main ENDP

CalNext PROC
         PUSH 0                                         ;push first number onto stack
         PUSH 1                                         ;push second number onto stack

         L1:
                   MOV ebx, 0                           ;clear ebx
                   MOV eax, 0                           ;clear eax

                   POP ax                               ;pop stack to eax
                   ADD bx, ax                           ;add popped number to bx
                   POP      ax                          ;pop stack to eax
                   ADD bx, ax                           ;add popped number to bx

                   MOV [esi + TYPE n], bx               ;store next FBNCI number in n

                   PUSH WORD PTR [esi]                  ;push previous number to stack
                   ADD esi, TYPE n                      ;increment esi
                   PUSH WORD PTR [esi]                  ;push new number to stack
         LOOP L1

         POP eax                                        ;pop extraneous data from stack
         POP eax                                        ;pop extraneous data from stack

         ret
CalNext ENDP

END main
```



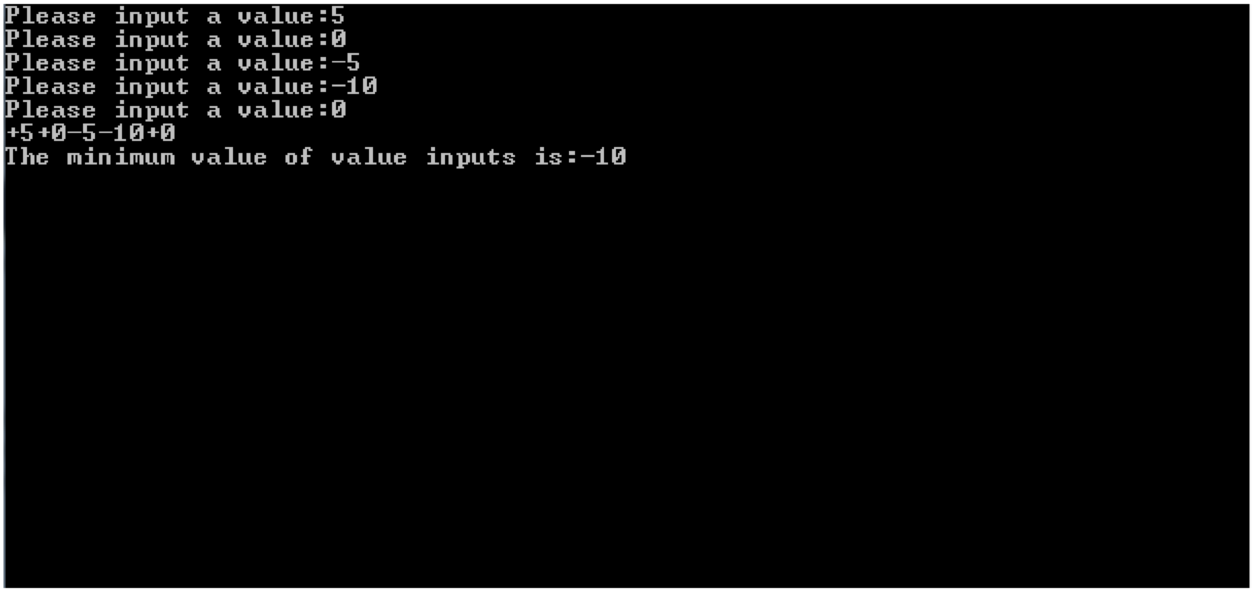
```
+0+1+1+2+3+5+8+13+21+34+55+89+144+233+377+610+987+1597+2584+4181
```

2. This question asks that you draft the `.code` portion of a program that will find the minimum value contained in an array. The `.data` portion of the program is provided below. Your completed program, containing both the `.data` and `.code` sections, must do the following:
   1) Prompt the user for integer input five times using the "`prompt`" string provided,
   2) Store each of the five user inputs in an array index,
   3) Find the minimum value, and;
   4) Display both the "`result`" string and the minimum value on the screen.

   Use the "call WriteInt" procedure, not "call DumpRegs". Establishing the minimum value can be done various ways, but must be done utilizing the "cmp" instruction. All submissions that evidence proper programming practice are acceptable. In your homework submission, please embed both the full program and one screen shot with at least one positive and one negative input value. ***Your program should NOT use procedures. This will be done in question #3.***

```
TITLE Array Assignment and Evaluation (main.asm)
; Description: Assigns user input to an array and finds the min value.
; Author: Matthew J Swann
; Version 1.0, 2012-08-02

INCLUDE Irvine32.inc
.data
prompt BYTE "Please input a value:", 0
result BYTE 0Dh, 0Ah, "The minimum value of value inputs is:",0
theArray DWORD 5 DUP(?)

.code
main PROC

        MOV ecx, 5
        MOV esi, OFFSET theArray
        MOV edx, OFFSET prompt

        L1:
                call WriteString
                call ReadInt
                MOV DWORD PTR [esi], eax
                ADD esi, TYPE theArray
        LOOP L1

        MOV ecx, LENGTHOF theArray
        MOV esi, OFFSET theArray

        L2:
                MOV eax, DWORD PTR [esi]
                ADD esi, TYPE theArray
                call WriteInt
        Loop L2

        MOV esi, OFFSET theArray
        MOV eax, DWORD PTR [esi]               ;move first number into eax
        MOV ecx, 4

        L3:
                ADD esi, TYPE theArray                  ;increment esi
                MOV ebx, DWORD PTR [esi]       ;move next number into ebx

                cmp eax, ebx                                      ;compare eax to ebx
                jg GREATER                                                 ;if eax > ebx, jump to GREATER

                dec ecx                                                    ;decrement ecx manually
                cmp ecx, 0
                jg L3                                                      ;manually call next iteration of L2 if ecx > 0
                jmp FINISH

                GREATER:
                        mov eax, ebx                              ;store new minimum in eax
        LOOP L3
```

```
          FINISH:

          MOV edx, OFFSET result
          call WriteString
          call WriteInt

          exit
main ENDP

END main
```



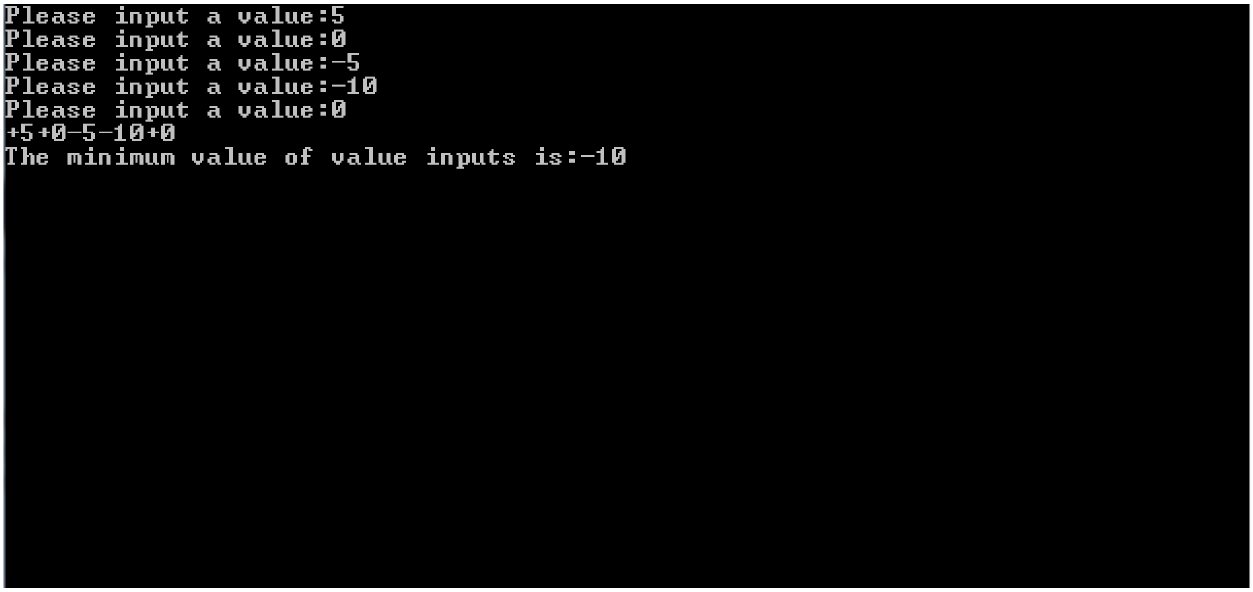
```
Please input a value:5
Please input a value:0
Please input a value:-5
Please input a value:-10
Please input a value:0
+5+0-5-10+0
The minimum value of value inputs is:-10
```

3. Re-draft the `.code` portion of the program from Question 2 using Procedures. The `.data` portion of the program is provided below. Your completed program, containing both the `.data` and `.code` sections, must adhere to the following:
   1) One procedure should prompt the user for input using the "`prompt`" string provided and store input into the array;
   2) A second procedure should print the contents of the array once filled;
   3) A third procedure should find the minimum value within the array; and,
   4) A final procedure should print the "`result`" string along with the minimum value.

   In your homework submission, please embed both the full program and one screen shot with at least one positive and one negative input value.

```
TITLE Array Assignment and Evaluation (main.asm)
; Description: Assigns user input to an array and finds the min value.
; Author: Matthew J Swann
; Version 1.0, 2012-08-02

INCLUDE Irvine32.inc
.data
prompt BYTE "Please input a value:", 0
result BYTE 0Dh, 0Ah, "The minimum value of value inputs is:",0
theArray DWORD 5 DUP(?)

.code
main PROC

         MOV ecx, 5
         MOV esi, OFFSET theArray
         MOV edx, OFFSET prompt

         L1:
                  call GetNum
         LOOP L1

         call PrintArray

         call FindMin

         call PrintResult

         exit
main ENDP

GetNum PROC
         call WriteString
         call ReadInt
         MOV DWORD PTR [esi], eax
         ADD esi, TYPE theArray

         ret
GetNum ENDP

PrintArray PROC
         MOV ecx, LENGTHOF theArray
         MOV esi, OFFSET theArray

         L1:
                  MOV eax, DWORD PTR [esi]
                  ADD esi, TYPE theArray
                  call WriteInt
         Loop L1

         ret
PrintArray ENDP

FindMin PROC
         MOV esi, OFFSET theArray
         MOV eax, DWORD PTR [esi]              ;move first number into eax
         MOV ecx, 4

         L1:
                  ADD esi, TYPE theArray                ;increment esi
                  MOV ebx, DWORD PTR [esi]     ;move next number into ebx
```

```
                   cmp eax, ebx                                   ;compare eax to ebx
                   jg GREATER                                              ;if eax > ebx, jump to GREATER

                   dec ecx                                                 ;decrement ecx manually
                   cmp ecx, 0
                   jg L1                                                   ;manually call next iteration of L2 if ecx > 0
                   jmp FINISH

                   GREATER:
                            mov eax, ebx                          ;store new minimum in eax
         LOOP L1

         FINISH:

         ret
FindMin ENDP

PrintResult PROC
         MOV edx, OFFSET result
         call WriteString
         call WriteInt

         ret
PrintREsult ENDP
END main
```



```
Please input a value:5
Please input a value:0
Please input a value:-5
Please input a value:-10
Please input a value:0
+5+0-5-10+0
The minimum value of value inputs is:-10
```