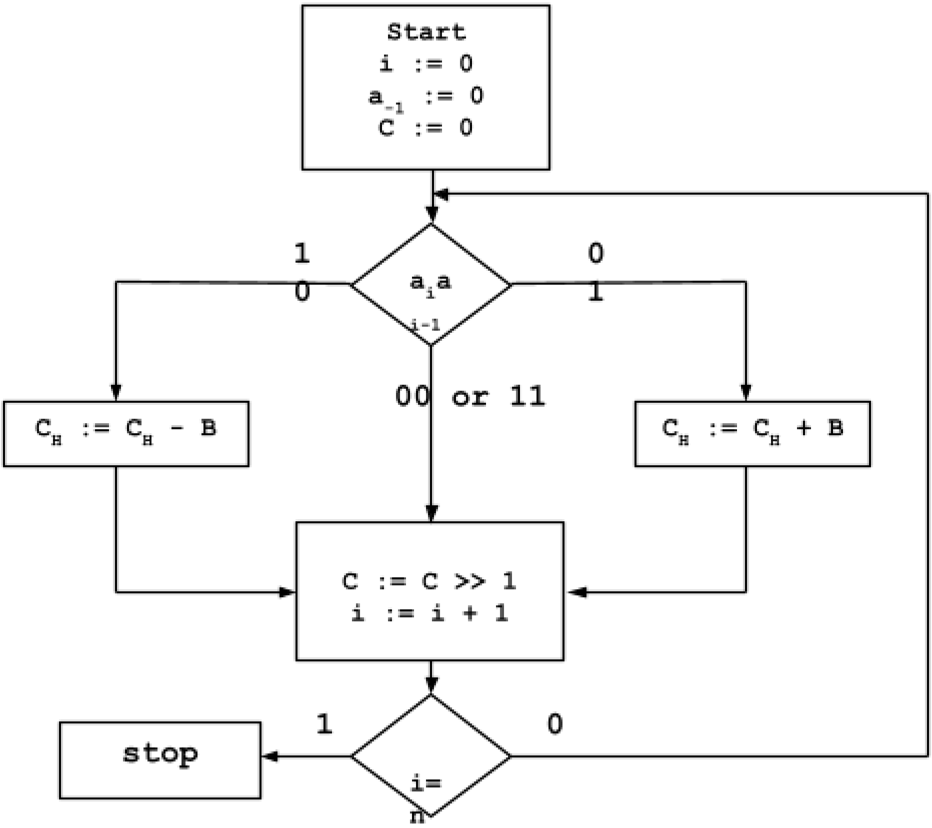
Homework #2

Name: Muhammed Orhun Gale
Student ID: 26754
Assigned: 29/03/2023
Due: 09/04/2023

1. (**30 pts**) Consider Booth's algorithm below for multiplying integers including signed ones (two' complement).



C = A × B, C is 2n-bit, A and B are n-bit registers. C_H is upper n-bit of C register.

Using Booth's algorithm with n = 4, do the following multiplication operations:

6 × 5 = ?
−5 × −4 = ?
−4 × 7 = ? (**10 pts each**)

Shows the steps of the algorithm.

6 × 5 = 00011110 (30)

i	B	A	a_ia_{i-1}	operation	C
0	0101	0110	00	C := C >> 1	00000000
1	0101	0110	10	Ch := Ch - B	10110000
				C := C >> 1	11011000
2	0101	0110	11	C := C >> 1	11011000
3	0101	0110	01	Ch := Ch + B	00111100
				C := C >> 1	00011110

-5 × -4 = 00010100 (20)

i	B	A	a_ia_{i-1}	operation	C
0	1100	1011	10	Ch = Ch - B	01000000
				C := C >> 1	00100000
1	1100	1011	11	C := C >> 1	00010000
2	1100	1011	01	Ch := Ch + B	11010000
				C := C >> 1	11101000
3	1100	1011	10	Ch = Ch - B	00101000
				C := C >> 1	00010100

-4 × 7 = 11100100 (-28)

i	B	A	a_ia_{i-1}	operation	C
0	0111	1100	00	C := C >> 1	00000000
1	0111	1100	00	C := C >> 1	00000000
2	0111	1100	10	Ch = Ch - B	10010000
				C := C >> 1	11001000
3	0111	1100	11	C := C >> 1	11100100

2. (**30 pts**) Consider the following C language statements.

- `f = -g + h + B[i]+ C[j]` (**10 pts**)
- `f = A[B[g]+ C[h] + j]` (**20 pts**)

Assume that the local variables `f, g, h, i` and `j` of integer types (32-bit) are assigned to registers `$s0, $s1, $s2, $s3` and `$s4` respectively. Assume also the base address of the arrays `A, B` and `C` of integer types are in registers `$s5, $s6` and `$s7`, respectively (i.e. `$s0 □ f, $s1 □ g, $s2 □ h, $s3 □ i, $s4 □ j, $s5 □ &A[0] , $s6 □ &B[0], $s7 □ &C[0]`).

For the C statements above, provide MIPS Assembly instructions.

1)

```
sll $t0, $s3, 2 #mult with 4
sll $t1, $s4, 2
add $t0, $t0, $s6 #get B[i]
add $t1, $t1, $s7 #get C[j]
lw  $s0, 0($t0) #Get values
lw  $t2, 0($t1)
add $s0, $s0, $t2 #Do arithmetic
add $s0, $s0, $s2
sub $s0, $s0, $s1
```

2)

```
sll  $t0, $s1, 2 #mult with 4
sll  $t1, $s2, 2
add  $t0, $t0, $s6 #get B[i]
add  $t1, $t1, $s7 #get C[j]
lw   $t2, 0($t0) #Get values
lw   $t1, 0($t1)
add  $t2, $t2, $t1 #Find location for vector A
add  $t2, $t2, $s4 #Lets say it is "k"
sll  $t2, $t2, 2
add  $t2, $t2, $s5 #Get A[k]
lw   $s0, 0($t2)
```

3. (**20 pts**) Consider the following C++ code sequence

```
t = A[0];
for (i=0; i < 5; i++)
     A[i] = A[i+1];
A[5] = t;
```

Write an Assembly language program for MIPS processor, assuming that base address of the array **A** is in **$s0**.

```
lw $t0, 0($s0)
add $t1, $zero, $zero
li $t2, 20

For:
     lw $t3, 4($t2)
     sw $t3, 0($t2)
     addi $t1, $t1, 4
     blt $t1, $t2, For

sw $t0, 20($s0)
```

4. (**20 pts**) Consider the following assembly program in MIPS, which implements a subroutine named "`func`"

```
# In: $a0 (an unsigned integer)
# Out: $v0

func:         add $s0, $zero, $a0
              addi $v0, $zero, 1

func loop:    beq $s0, $zero, func return
              add $v0, $v0, $s0
              addi $s0, $s0, -1
              j func loop

func return:jr $r
```

a. (**15 pts**) What is the C/C++ equivalent of this code? Assume that the functions argument is an unsigned integer `n` in the C/C++ version of the function. What is the value returned by this function if it is called with `$a0=5`?

```
int func(int n){
     int v = 1;
     while(n != 0)
     {
          v = v + n;
          n = n - 1;
     }
     return v;
}
```

```
1) n = 5 -> v = 6
2) n = 4 -> v = 10
3) n = 3 -> v = 13
4) n = 2 -> v = 15
5) n = 1 -> v = 16
6) n = 0 -> v = 16 → return 16
```

b. (**5 pts**) The code in the box above contains an unconventional use of registers that violates the MIPS procedure calling convention. This may result in an error. Find this unconventional use of registers and show how it should be fixed.

jr should return with $ra register