RISC-V Problem 1 - Given the mapping of registers to variables below, write a program to implement the following expression:

```
int func(int A, int B, int C, int D){
      return (A+B)*(C-D);
}
int main(){
      int A=4, B=2, C=6, D=3;
      int z = func(A, B, C, D);
      return 0;
}
```

Registers in main are A=x8, B=9, C=x18, D=x19, Z=x20

You are permitted to use labels in this problem

```
MAIN:

addi x8, x0, 4
addi x9, x0, 2
addi x18, x0, 6
addi x19, x0, 3

# 1) Use the function argument reg to pass by value
add x12, x8, x0  # x12 is A passed by value
add x13, x9, x0  # x13 is B passed by value
add x14, x18, x0  # x14 is C passed by value
add x15, x19, x0  # x15 is D passed by value

# 2) Call the function
jal x1, MUL_FUNC

# 3) Store the result in the main function result
add x20, x10, x0

# 4) Return 0
beq x0, x0, END
```

```
# 5) Name the function call
MUL_FUNC:

# 6) Add the passed by value copies
add x5, x12, x13
sub x6, x14, x15

add x7, x0, x0   # Iterator is in x7
add x28, x0, x0  # Multiplication Sum is x28

# 7) Loop
MUL_LOOP:
sub x29, x7, x6  # We stop when i - x6 = 0
beq x29, x0, PRINT_SUM  # Branch to print sum when done
add x28, x28, x5
addi x7, x7, 1
beq x28, x28, MUL_LOOP

# 8) Put the result in one of the return registers
PRINT_SUM:
add x10, x28, x0

# 9) Jump and link back to main
jalr x0, x1, 0

# 10) Terminate the program
END:
quit
```

RISC-V Problem 2 - Given the mapping of registers to variables below, write a program to implement the following expression:

```
void swap(int* A, int* B){
      int temp = *A;
      *A = *B;
      *B = temp;
}

int main(){
      int A=10, B=7;
      swap(&A, &B);
      return 0;
}
```

Registers in main are A=x18, B=19
For simplicity, store A at 0x7fff1a10 for pass by reference
For simplicity, store B at 0x7fff1a18 for pass by reference

You are permitted to use labels in this problem

```
# Starter Code
MAIN:

addi x18, x18, 10
addi x19, x19, 17

# Write the Solution Here
# 1) Use the function argument registers to
pass by value
lui x10, 0x7fff1
lui x11, 0x7fff1
addi x10, x10, 0xa10
addi x11, x11, 0xa18
sw x18, 0(x10)
sw x19, 0(x11)

# Call the function
jal x1, SWAP_FUNC

# Load the results back into the local
registers
lw x18, 0(x10)
lw x19, 0(x11)

# Return 0
beq x0, x0, END
```

```
# Name the function call
SWAP_FUNC:

# int temp = *A;
lw x5, 0(x10)

# *A = *B;
lw x6, 0(x11)
sw x6, 0(x10)

# *B = temp;
sw x5, 0(x11)

# Jump and link back to main
jalr x0, x1, 0

# Terminate the program
END:
Quit
```

RISC-V Problem 3 - Given the mapping of registers to variables below, write a program to implement the following expression:

```
int array_sum(int* A, int length){

        int sum = 0;
        for(int i = 0; i < length; ++i){
                sum += A[i];
        }
        return sum;
}

int main(){
        int array[] = {13, 7, -8, 4};
        int array_len = 4;
        int result_sum = array_sum( array, array_len );
        return 0;
}
```

In main, array is in x18 and points to 0x7fff1a10, array_len is in x19, and result_sum is in x20. In array_sum, sum is in x21 and i is in x22.

```
MAIN:

lui x5, 0x7fff1
addi x18, x5, 0xa10

# Store the array elementd
addi x6, x0, 13
addi x7, x0, 7
addi x28, x0, -8
addi x29, x0, 4

sw x6, 0(x18)
sw x7, 4(x18)
sw x28, 8(x18)
sw x29, 12(x18)

# Store the length
addi x19, x0, 5

# Put in function inputs
addi x10, x18, 0
addi x11, x19, 0

# Call the function
jal x1, ARRAY_SUM

# Load the results back into the local
registers
add x20, x12, x0

# Return 0
beq x0, x0, END
```

```
# Name the function call
ARRAY_SUM:

add x21, x0, x0
add x22, x0, x0

BEGIN_LOOP:
beq x22, x11, END_LOOP
slli x6, x22, 2
add x7, x10, x6
lw x28, 0(x7)
add x21, x21, x28
addi x22, x22, 1
beq x0, x0, BEGIN_LOOP

# Jump and link back to main
END_LOOP:

# Return result
add x12, x21, x0

# Go back to main
jalr x0, x1, 0

# Terminate the program
END:
quit
```