1. LOOP (i)

```
{
        function[i]

}
```

2. function containing loop that is running for every i

I thought it would be interesting to see how these two different implementations of the same functionality behave in terms of efficiency. If we judge the efficiency of the system solely on memory accesses, then it turns out that the number of memory accesses performed(sw,lw) is actually the same for both cases.

<u>for loop outside of function:</u>

```
#define N 8



void fill_array_with_addition (int *A,int *B,int *C)
{



        *C=*A+*B;



}

int main (void)
{
    int A[N]={ 7 , 3 , 25 , 4 , 75 , 2 , 1 , 1 };
    int B[N]={ 3 , 7 , 25 , 6 , 25 , 8 , 9 , 111 };
    int C[N];
     int i;
```

```
        for (i=0; i<N; i++)
        {
        fill_array_with_addition(&A[i],&B[i],&C[i]);
        }




    return 0;
}
```

```
# FOR LOOP OUTSIDE

0x0000009a: 59 71              addi  sp,sp,-112
0x0000009c: 86 d6              sw    ra,108(sp)
0x0000009e: a2 d4              sw    s0,104(sp)
0x000000a0: 93 07 40 18        li    a5,388
0x000000a4: 03 a3 07 00        lw    t1,0(a5)
0x000000a8: 83 a8 47 00        lw    a7,4(a5)
0x000000ac: 03 a8 87 00        lw    a6,8(a5)
0x000000b0: c8 47              lw    a0,12(a5)
0x000000b2: 8c 4b              lw    a1,16(a5)
0x000000b4: d0 4b              lw    a2,20(a5)
0x000000b6: 94 4f              lw    a3,24(a5)
0x000000b8: d8 4f              lw    a4,28(a5)
0x000000ba: 9a c0              sw    t1,64(sp)
0x000000bc: c6 c2              sw    a7,68(sp)
0x000000be: c2 c4              sw    a6,72(sp)
0x000000c0: aa c6              sw    a0,76(sp)
0x000000c2: ae c8              sw    a1,80(sp)
0x000000c4: b2 ca              sw    a2,84(sp)
0x000000c6: b6 cc              sw    a3,88(sp)
0x000000c8: ba ce              sw    a4,92(sp)
0x000000ca: 83 a8 07 02        lw    a7,32(a5)
0x000000ce: 03 a8 47 02        lw    a6,36(a5)
0x000000d2: 88 57              lw    a0,40(a5)
0x000000d4: cc 57              lw    a1,44(a5)
0x000000d6: 90 5b              lw    a2,48(a5)
0x000000d8: d4 5b              lw    a3,52(a5)
0x000000da: 98 5f              lw    a4,56(a5)
0x000000dc: dc 5f              lw    a5,60(a5)
```

```
0x000000de: 46 d0              sw    a7,32(sp)
0x000000e0: 42 d2              sw    a6,36(sp)
0x000000e2: 2a d4              sw    a0,40(sp)
0x000000e4: 2e d6              sw    a1,44(sp)
0x000000e6: 32 d8              sw    a2,48(sp)
0x000000e8: 36 da              sw    a3,52(sp)
0x000000ea: 3a dc              sw    a4,56(sp)
0x000000ec: 3e de              sw    a5,60(sp)
0x000000ee: 01 44              li    s0,0
0x000000f0: 21 a8              j     0x108 <main+110>
0x000000f2: 13 15 24 00        slli  a0,s0,0x2
0x000000f6: 33 06 a1 00        add   a2,sp,a0
0x000000fa: 1c 10              addi  a5,sp,32
0x000000fc: b3 85 a7 00        add   a1,a5,a0
0x00000100: 9c 00              addi  a5,sp,64
0x00000102: 3e 95              add   a0,a0,a5
0x00000104: 71 37              jal   0x90 <fill_array_with_addition>
0x00000106: 05 04              addi  s0,s0,1
0x00000108: 9d 47              li    a5,7
0x0000010a: e3 d4 87 fe        bge   a5,s0,0xf2 <main+88>
0x0000010e: 01 45              li    a0,0
0x00000110: b6 50              lw    ra,108(sp)
0x00000112: 26 54              lw    s0,104(sp)
0x00000114: 65 61              addi  sp,sp,112
0x00000116: 82 80              ret
# 36 memory accesses

# fill_array_with_addition

0x00000090: 1c 41              lw    a5,0(a0)
0x00000092: 98 41              lw    a4,0(a1)
0x00000094: ba 97              add   a5,a5,a4
0x00000096: 1c c2              sw    a5,0(a2)
0x00000098: 82 80              ret


# 3 memory accesses * 8 = 24
```

Total memory accesses =60

<u>for loop inside function</u>

```c
#define N 8



void fill_array_with_addition (int A[],int B[],int C[])
{
    int i;

        for (i=0; i<N; i++)
    {

        C[i]=A[i]+B[i];

    }

}

int main (void)
{
    int A[N]={ 7 , 3 , 25 , 4 , 75 , 2 , 1 , 1 };
    int B[N]={ 3 , 7 , 25 , 6 , 25 , 8 , 9 , 111 };
    int C[N];
    fill_array_with_addition(A,B,C);





    return 0;
}
```

```
# testing loops

# main
0x000000b6: 59 71               addi  sp,sp,-112
0x000000b8: 86 d6               sw    ra,108(sp)
0x000000ba: 93 07 40 18         li    a5,388
0x000000be: 03 a3 07 00         lw    t1,0(a5)
0x000000c2: 83 a8 47 00         lw    a7,4(a5)
0x000000c6: 03 a8 87 00         lw    a6,8(a5)
0x000000ca: c8 47               lw    a0,12(a5)
0x000000cc: 8c 4b               lw    a1,16(a5)
```

```
0x000000ce: d0 4b              lw    a2,20(a5)
0x000000d0: 94 4f              lw    a3,24(a5)
0x000000d2: d8 4f              lw    a4,28(a5)
0x000000d4: 9a c0              sw    t1,64(sp)
0x000000d6: c6 c2              sw    a7,68(sp)
0x000000d8: c2 c4              sw    a6,72(sp)
0x000000da: aa c6              sw    a0,76(sp)
0x000000dc: ae c8              sw    a1,80(sp)
0x000000de: b2 ca              sw    a2,84(sp)
0x000000e0: b6 cc              sw    a3,88(sp)
0x000000e2: ba ce              sw    a4,92(sp)
0x000000e4: 83 a8 07 02        lw    a7,32(a5)
0x000000e8: 03 a8 47 02        lw    a6,36(a5)
0x000000ec: 88 57              lw    a0,40(a5)
0x000000ee: cc 57              lw    a1,44(a5)
0x000000f0: 90 5b              lw    a2,48(a5)
0x000000f2: d4 5b              lw    a3,52(a5)
0x000000f4: 98 5f              lw    a4,56(a5)
0x000000f6: dc 5f              lw    a5,60(a5)
0x000000f8: 46 d0              sw    a7,32(sp)
0x000000fa: 42 d2              sw    a6,36(sp)
0x000000fc: 2a d4              sw    a0,40(sp)
0x000000fe: 2e d6              sw    a1,44(sp)
0x00000100: 32 d8              sw    a2,48(sp)
0x00000102: 36 da              sw    a3,52(sp)
0x00000104: 3a dc              sw    a4,56(sp)
0x00000106: 3e de              sw    a5,60(sp)
0x00000108: 0a 86              mv    a2,sp
0x0000010a: 0c 10              addi  a1,sp,32
0x0000010c: 88 00              addi  a0,sp,64
0x0000010e: 49 37              jal   0x90 <fill_array_with_addition>
            # around 36 mem accesses
0x00000110: 01 45              li    a0,0
0x00000112: b6 50              lw    ra,108(sp)
0x00000114: 65 61              addi  sp,sp,112
0x00000116: 82 80              ret




# fill array function



0x00000090: 01 47              li    a4,0
0x00000092: 31 a8              j     0xae <fill_array_with_addition+30>
0x00000094: 93 17 27 00        slli  a5,a4,0x2
```

```
0x00000098: b3 06 f5 00       add   a3,a0,a5
0x0000009c: 94 42             lw    a3,0(a3)
                  # mem access
0x0000009e: 33 88 f5 00       add   a6,a1,a5
0x000000a2: 03 28 08 00       lw    a6,0(a6) # mem access

0x000000a6: b2 97             add   a5,a5,a2
0x000000a8: c2 96             add   a3,a3,a6
0x000000aa: 94 c3             sw    a3,0(a5) # mem access
0x000000ac: 05 07             addi  a4,a4,1
0x000000ae: 9d 47             li    a5,7
0x000000b0: e3 d2 e7 fe       bge   a5,a4,0x94
<fill_array_with_addition+4>  # 3 mem accesses * 8=24
0x000000b4: 82 80             ret

# total = 24+36=60
```

Total memory accesses =60