EE234

Microprocessor Systems

Final Exam

Dec. 12, 2023. (1:10pm – 4:00pm)

Instructor: Dae Hyun Kim (daehyun@eecs.wsu.edu)

Name:

WSU ID:

Problem	Points	
1	20	
2	20	
3	40	
4	20	
5	20	
6	20	
Total	140	

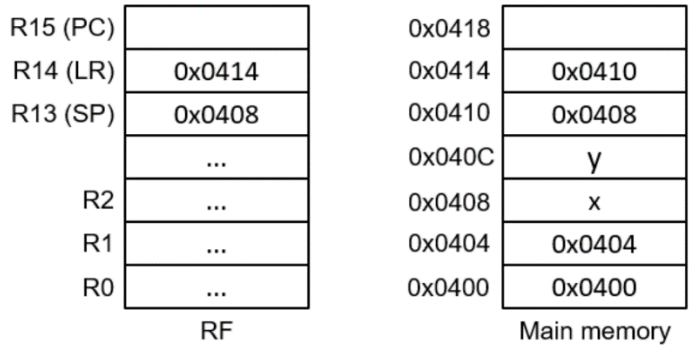
Problem #1 (Array, 20 points)

You can use the following instructions only in this exam.

- Instructions
  - ADD, SUB, AND, ORR, EOR, MOV, MUL
  - CMP, BGE/BLT/BGT/BLE/BEQ/BNE
  - B, BL, BX
  - LDR, STR, PUSH, POP

Write an assembly code for the "for loop" in the following C code.

```
int* x = new int[10];
int* y = new int[10];
...
for ( int k = 0 ; k < 10 ; k++ ) {
  int a = x[k];
  x[k] = y[k];
  y[k] = a;
}
```



- R0-R12 are freely available.
- You can use any of R0-R12 for "int k" and "int a" (i.e., you don't need to use the stack for k and a).

```
  MOV R0, #0
loop:
  CMP R0, #10
  BGE end
  LDR R1, [SP]
  MUL R2, R0, #4
  ADD R2, R2, R1
  LDR R3, [R2]
  LDR R4, [SP, #4]
  MUL R5, R0, #4
  ADD R5, R5, R4
  LDR R6, [R5]
  STR R6, [R2]
  STR R3, [R5]
  ADD R0, R0, #1
  B loop
end:
```

Problem #2 (Array, 20 points)

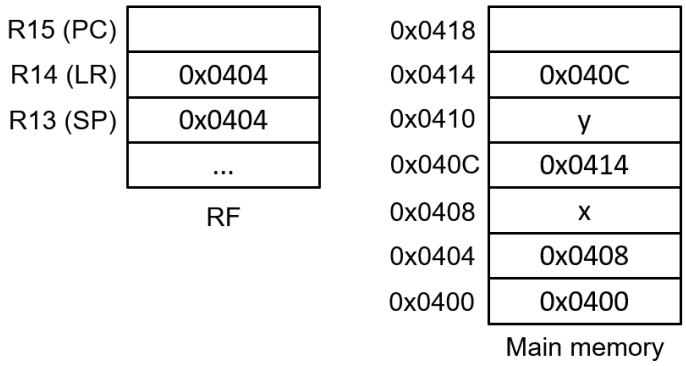
Write an assembly code for the second "for loop" (the one in the red rectangle) in the following C code. A "long" variable occupies eight bytes.

```
long** x = new long*[20];
long** y = new long*[20];

for ( int k = 0 ; k < 20 ; k++ ) {
  x[k] = new long[20];
  y[k] = new long[20];
}

...

for ( int m = 0 ; m < 10 ; m++ ) {
  y[m][2*m] = x[2*m][m];
}
```



- R0-R12 are freely available.
- You can use any of R0-R12 for "int m" (i.e., you don't need to use the stack for "m").

```
  MOV R0, #0
loop:
  CMP R0, #10
  BGE end
  LDR R1, [SP, #4]
  MUL R2, R0, #2
  MUL R2, R2, #4
  ADD R2, R2, R1
  LDR R3, [R2]
  MUL R2, R0, #8
  ADD R2, R2, R3
  LDR R3, [R2]
  LDR R4, [R2, #4]
  LDR R5, [SP, #12]
  MUL R6, R0, #4
  ADD R6, R6, R5
  LDR R7, [R6]
  MUL R8, R0, #2
  MUL R8, R8, #8
  ADD R8, R8, R7
  STR R3, [R8]
  STR R4, [R8, #4]
  ADD R0, R0, #1
  B loop
end:
```

Problem #3 (Array, 40 points)

(1) How many bytes does the array "a" actually use for the array? (6 points) 40B

```
int a[10];
```

(2) How many bytes does the array "b" actually use for the array? (6 points) 44B

```
int* b = new int[10];
```

(3) How many bytes does the array "c" actually use for the array? (6 points) 400B

```
int c[10][10];
```

(4) How many bytes does the array "d" actually use for the array? (6 points) 444B

```
int** d = new int*[10];
for ( int m = 0 ; m < 10 ; m++ ) {
  d[m] = new int[10];
}
```

(5) How many bytes does the array "e" actually use for the array? (6 points) 844B

```
long** e = new long*[10];
for ( int m = 0 ; m < 10 ; m++ ) {
  e[m] = new long[10];
}
```

(6) We define a 5-dimensional dynamic array of long variables as follows. How many bytes does the array "g" actually use for the array? n1, n2, n3, n4, n5 are constants and your answer should be expressed as a function of those constants. (10 points) n1*n2*n3*n4*n5*8 + 4 + 4*n1 + 4*n1*n2 + 4*n1*n2*n3 + 4*n1*n2*n3*n4

```
long***** g = new long****[n1];
for ( int k1 = 0 ; k1 < n1 ; k1++ ) {
  g[k1] = new long***[n2];
  for ( int k2 = 0 ; k2 < n2 ; k2++ ) {
    g[k1][k2] = new long**[n3];
    for ( int k3 = 0 ; k3 < n3 ; k++ ) {
      g[k1][k2][k3] = new long*[n4];
      for ( int k4 = 0 ; k4 < n4 ; k++ ) {
        g[k1][k2][k3][k4] = new long[n5];
      }
    }
  }
}
```

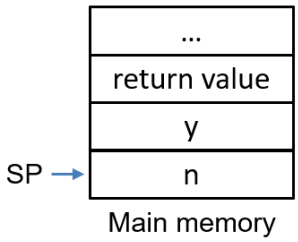
Problem #4 (Array, 20 points)

Write an assembly code for the com() function.

```
int com (int* y, int n) {
  int s = 0;

  for ( int k = 0 ; k < n ; k++ ) {
    s = s + *(y+k);
  }

  return s;
}
```



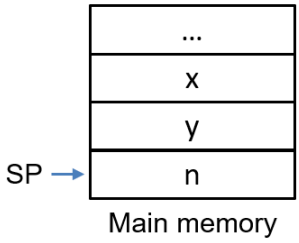
- R0-R12 are freely available.
- You can use any of R0-R12 for “int s” and “int k” in the com() function.
- Use the stack memory for the function arguments and the return value (see the figure).

```
  MOV R0, #0
  LDR R1, [SP]
  LDR R2, [SP, #4]
  MOV R3, #0
loop:
  CMP R3, R1
  BGE end
  MUL R4, R3, #4
  ADD R4, R4, R2
  LDR R4, [R4]
  ADD R0, R0, R4
  ADD R3, R3, #1
  B loop
end:
  STR R0, [SP, #8]
  BX LR
```

Problem #5 (Array, 20 points)

Write an assembly code for the com() function. A "long" variable occupies 8 bytes.

```
void com (long* x, long* y, int n) {
  for ( int k = 0 ; k < n ; k++ ) {
    x[k] = y[n-k-1];
  }
}
```



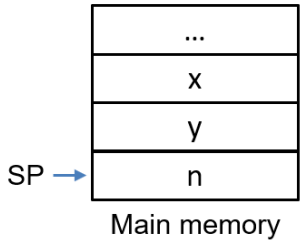
- R0-R12 are freely available.
- You can use any of R0-R12 for "int k" in the com() function.
- Use the stack memory for the function arguments and the return value (see the figure).

```
  LDR R1, [SP]
  LDR R2, [SP, #4]
  LDR R3, [SP, #8]
  MOV R0, #0
loop:
  CMP R0, R1
  BGE end
  SUB R4, R1, #1
  SUB R4, R4, R0
  MUL R4, R4, #8
  ADD R4, R4, R2
  LDR R5, [R4]
  LDR R6, [R4, #4]
  MUL R4, R0, #8
  ADD R4, R4, R3
  STR R5, [R4]
  STR R6, [R4, #4]
  ADD R0, R0, #1
  B loop
end:
  BX LR
```

Problem #6 (Array, 20 points)

Write an assembly code for the com() function. A “long” variable occupies 8 bytes.

```
void com (long** x, long** y, int n) {
  for ( int k = 0 ; k < n ; k++) {
    for ( int s = 0 ; s < n ; s++ ) {
      x[k][s] = y[n-s-1][n-k-1]
    }
  }
}
```



- R0-R12 are freely available.
- You can use any of R0-R12 for “int k” and “int s” in the com() function.
- Use the stack memory for the function arguments and the return value (see the figure).

```
  LDR R2, [SP]
  LDR R3, [SP, #4]
  LDR R4, [SP, #8]
  MOV R0, #0
loop1:
  CMP R0, R2
  BGE loop1_end
  MOV R1, #0
loop2:
  CMP R1, R2
  BGE loop2_end
  SUB R5, R2, R1
  SUB R5, R5, #1
  MUL R5, R5, #4
  ADD R5, R5, R3
  LDR R5, [R5]
  SUB R6, R2, R0
  SUB R6, R6, #1
  MUL R6, R6, #8
  ADD R6, R6, R5
  LDR R8, [R6]
  LDR R9, [R6, #4]
  MUL R5, R0, #4
  ADD R5, R5, R4
  LDR R5, [R5]
  MUL R6, R1, #8
  ADD R6, R6, R5
  STR R8, [R6]
  STR R9, [R6, #4]
  ADD R1, R1, #1
  B loop2
loop2_end:
  ADD R0, R0, #1
  B loop1
loop1_end:
  BX LR
```