C to MIPS (If Statements) Practice Exercises

Students, it is extremely important (for the learning process) for you to try and solve these questions before viewing the answers.

1) Try to translate the following to MIPS code, where i maps to $s0, and j maps to $s1 (the solution is provided at the end of the document)

```
if (i == j)
        i = i + 1;
j = j - 1;
```

2) Try to translate the following to MIPS code, where i maps to $s0, and j maps to $s1 (the solution is provided on the 2nd and 3rd pages of the document)

```
if (i == j)
        i = i + 1;
else
        j = j - 1;
j = j + i;
```

3) Try to translate the following to MIPS code, where i maps to $s0, and j maps to $s1 (the solution is provided at the end of the document)

```
if ((i == j) && (i == k))              \\ && is the AND logical operator
        i = i + 1;
else
        j = j - 1;
j = i + k;
```

4) Try to translate the following to MIPS code, where i maps to $s0, and j maps to $s1 (the solution is provided at the end of the document)

```
if ((i == j) || (i == k))              \\ || is the OR logical operator
        i = i + 1;
else
        j = j - 1;
j = i + k;
```

Solution to Question 1 (i maps to $s0, and j maps to $s1):

C Code:

```
if (i == j)
        i = i + 1;
j = j - 1;
```

```
bne  $s0, $s1, L1        # branch if ( i != j )
addi $s0, $s0, 1         # i = i + 1
L1: addi $s1, $s1, -1    # j = j - 1
```

or

```
bne  $s0, $s1, L1    # branch if ( i != j )
addi $s0, $s0, 1     # i = i + 1
L1:
addi $s1, $s1, -1    # j = j - 1
```

Solution to Question 2 (i maps to $s0, and j maps to $s1):

C Code:

```
if (i == j)
        i = i + 1;
else
        j = j - 1;
j = j + i;
```

```
bne  $s0, $s1, ELSE   # branch if  ( i != j )
addi $s0, $s0, 1      # i = i + 1
j L1                  # jump over else
ELSE:
addi $s1, $s1, -1     # j = j - 1
L1:
add $s1, $s1, $s0     # j = j + i
```

Solution to Question 3 (i maps to $s0, j maps to $s1, and k maps to $s2):

```
if ((i == j) && (i == k))              \\ && is the AND logical operator
        i = i + 1;
else
        j = j - 1;
j = i + k;
```

```
bne  $s0, $s1, ELSE   # condition 1: branch if ( i != j )
bne  $s0, $s2, ELSE   # condition 2: branch if ( i != k )
addi $s0, $s0, 1      # if-body: i = i + 1
j L1                  # jump over else
ELSE:
addi $s1, $s1, -1     # else-body: j = j - 1
L1:
add $s1, $s0, $s2      # j = i + k
```

Solution to Question 4 (i maps to $s0, j maps to $s1, and k maps to $s2):

```
if ((i == j) || (i == k))          \\ || is the OR logical operator
        i = i + 1;
else
        j = j - 1;
j = i + k;
```

```
beq  $s0, $s1, IF       # condition 1: branch if ( i == j )
bne  $s0, $s2, ELSE     # condition 2: branch if ( i != k )
IF:
addi $s0, $s0, 1        # if-body: i = i + 1
j L1                    # jump over else
ELSE:
addi $s1, $s1, -1       # else-body: j = j - 1
L1:
add $s1, $s0, $s2       # j = i + k
```