ICS-365 HW10

\* Due Date – See Class Schedule \*

1. Rewrite the following pseudocode segment using a loop structure in the specified languages:

k = (j + 13) / 27

loop:

**if** k > 10 **then goto** out

k = k + 1

i = 3 \* k - 1

**goto** loop

out: . . .

1. C

k = (j + 13) / 27;

while( k <= 10)

{

k++;

i = 3 \* k - 1;

}

1. Python

k = (j + 13) / 27

while k <= 10;

k += 1

i = 3 \* k - 1

1. Rewrite the following code segment using a multiple-

Selection statement in the following languages:

**if** ((k == 1) || (k == 2)) j = 2 \* k - 1

**if** ((k == 3) || (k == 5)) j = 3 \* k + 1

**if** (k == 4) j = 4 \* k - 1

**if** ((k == 6) || (k == 7) || (k == 8)) j = k – 2

1. C or Java

switch(k)

{

case 1: case 2:

j = 2 \* k -1;

break;

case 3: case 5:

j = 3 \* k + 1;

break;

case 4:

j = 4 \* k - 1;

break;

case 6: case 7: case 8:

j = k - 2;

break:

default:

printf(“Error!”);

break;

}  
  
b. Python

if (k == 1) or (k == 2):

j == 2 \* k – 1

elif (k == 3) or (k == 5):

j = 3 \* k + 1

elif (k == 4):

j = 4 \* k – 1

elif (k == 6) or (k == 7) or (k == 8):

j = k - 2

else:

print(k, ‘ - Error in case k value’)

Assume all variables are integer type. Discuss the relative merits of

the use of these languages for this particular code.

1. Consider the following C program segment. Rewrite it using no gotos or **break**s.

j = -3;

for (i = 0; i < 3 && j <= 0; i++) {

if (j + 2 == 3 || j + 2 == 2) {

j--;

} else if (j + 2 == 0) {

j += 2;

} else {

j = 0;

}

if (j <= 0) {

j = 3 – i;

}

}

1. Consider the following programming problem: The values of three

Integer variables—first, second, and third—must be placed in the

three variables max, mid, and min, with the obvious meanings, without using arrays or user-defined or predefined subprograms. Write two solutions to this problem, one that uses nested selections and one that does not. Compare the complexity and expected reliability of the two.

The one with nested if will work a little faster because it has less compares to do and it will also be a little more reliable (it has less if statements and less code), but it is harder to read.

Code on next page:

Nested ifs:

if (first >= second) {

if (first >= third) {

max = first;

if (second >= third) {

mid = second;

min = third;

} else {

mid = third;

min = second;

}

} else {

max = third;

mid = first;

min = second;

}

} else if (second >= third) {

max = second;

if (first >= third) {

mid = first;

min = third;

} else {

mid = third;

min = first;

}

} else {

max = third;

mid = second;

min = first;

}

No nested ifs:

if (first >= second && second >= third) {

max = first;

mid = second;

min = third;

} else if (first >= third && third >= second) {

max = first;

mid = third;

min = second;

} else if (second >= third && third >= second) {

max = third;

mid = first;

min = second;

} else if (third >= second && second >= first) {

max = third;

mid = second;

min = first;

} else if (second >= first && first >= third) {

max = second;

mid = first;

min = third

} else if (second >= third && third >= first) {

max = second;

mid = third;

min = first;

}