**Homework 3**

**100**

**Name: \_\_\_\_Devon Layton\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (Total points: 100)**

**Requirement: All answers (except parse trees) must be typed.**

1. Have you read the 2nd exam guidance document in D2L under “Exam Guidance” folder? (10 point if your answer is positive).

**Positive** (Yes, I did)

1. Rewrite the following code segment using a multiple-selection statement (**switch or case expression, Not nested if statements**) in the following languages: (20 points. 10 for each)



* C#

switch(k) {

case 1:

case 2: j = 2 \* k – 1;

break;

case 3: j = 3 \* k + 1;

break;

case 4: j = 4 \* k – 1;

break;

case 5: j = 3 \* k + 1;

break;

case 6:

case 7:

case 8: j = k – 2;

break;

default: break;

}

* Ruby

case k

when 1, 2 puts j = 2 \* k – 1

when 3, 5 puts j = 3 \* k + 1

when 4 puts j = 4 \* k – 1

when 6,7,8 puts j = k – 2

end

1. Let the function fun be defined as



What are the values of sum1 and sum2? (20 points, 10 for each)

* if the operands in the expressions are evaluated left to right?

sum1: 46 sum2: 48 Left🡪Right

Sum1: (10/2) + 41 = 46

Sum2: 41 + (14/2) = 48

* if the operands in the expressions are evaluated right to left?

sum1: 48 sum2: 46 Right🡪Left

Sum1: 41 + (14/2) = 48

Sum2: (10/2) + 41 = 46

1. Please define *functional side effect*. (10 points)

Functional side effect occurs when some state change is caused by executing a function. The function has changed a two-way parameter or non-local variable which then gets used elsewhere (other functions or calculations), creating undesired effects in the program.

1. State your own arguments for and against allowing mixed-mode arithmetic expressions (i.e. operands are of different types). (10 points)

For: Mixed-mode arithmetic is beneficial to preserve information and more precise, accurate calculations because there are times when various data has different types. It allows the programmer to write programs that need to perform more complex calculations.

Against: Although the writability is improved with mix-mode arithmetic, the overall reliability is hurt because calculations may not be what the programmer anticipated. Since type inference determines which data type the operands are converted to, it may result in an unexpected outcome and may be difficult to identify where the problem is occurring.

1. What is short-circuit evaluation? (10 points)

Short-circuit evaluation iterates through an expression of Boolean conditions(subexpressions). The first Boolean condition that is satisfied, determines the outcome/value of the expression and stops evaluation before any other conditions get evaluated.

1. Please specify the differences of ***break*** statement in C++ and Java. (10 points)

C++ break statements terminate the innermost loop (for, while, etc.) it is in, skips the current iteration and continues to the next iteration. Java’s break is different in that you can have a break followed by a label to specify which loop to terminate. Java does support both labeled and unlabeled (behaves like other languages) breaks whereas C++ breaks are unlabeled.

1. Assume the following rules of associativity and precedence for expressions:



Show the order of evaluation of the following expressions by parenthesizing

all subexpressions and placing a superscript on the right parenthesis

to indicate order. For example, for the expression

a + b \* c + d

the order of evaluation would be represented as

((a + (b \* c)1 )2 + d)3 (10 points. 5 for each)

1. (((a \* b)1 – 1)2 + c)3
2. (((a \* (b - 1)1)2 / c)3 **mod** d)4