Dara Lim

CS4250

Homework#4

1. Th ‘add’ operator in a programming language would not be commutative in case of operations performed on strings and user-defined datatypes. Consider the following example’s result of the concatenation operation “Programming” + “Language” is “ProgrammingLanguage”, but the result of “Language” + “Programming” is “LanguageProgramming”. The result is not the same in case the strings are concatenated in different order.

2. The ‘add’ operator in a programming language would not be associative in the following example: Suppose that a program must evaluate the expression A + B + C + D and that A and C are very large positive numbers, and B and D are negative numbers with very large absolute values. In this situation, adding B to A does not cause an overflow exception, but adding C to A does. Likewise, adding C to B does not cause overflow, but adding D to B does. Because of the limitations of computer arithmetic, addition is catastrophically non-associative in this case.

3. It is difficult to eliminate functional side effects in C because in C there is only function, which mean that all subprograms only return one value. One way to eliminate the side effect of the two way parameter and still provide subprogram return more than one value, the values would need to be place in a struct and the struct returned. Access to global in functions would also have to be disallowed.

4. a. Operands in the expressions are evaluated left to right:

-The value of sum1 is: **34**

-The value of sum2 is: **46**

b. Operands in the expressions are evaluated right to left:

-The value of sum1 is: **46**

-The value of sum2 is: **34**

5. a. Operands are evaluated left to right, the value of x is: **7**

b. Operands are evaluated right to left, the value of x is: **12**