Quiz 02 (5 points + 1 bonus, 30mins)

Name: WonpyoHong Student ID: 21701065 Section #: TF5

Open book quiz: You can refer to any reference including google search. But do not copy and paste but answer in your own words. Do not discuss with other people.

You can take this quiz either online or offline.

Online Quiz: Copy this. [File] menu >> [Make a copy menu] >> Put it in your shared PL class folder (Name your file, Quiz02-YourStudentID-Name)

Offline Quiz: Just write your answer in the printed sheet of this file. (Students in the class may take this as an online quiz as well.)

1. Discuss briefly the benefit of the parser-interpreter architecture when designing and implementing a new programming language.(1 point) **(**Bonus +1**: Suggest a better architecture to design and implement a programming language than the parser-interpreter architecture.)**

The biggest benefit of parser-interpreter architecture is that it can be used universally. Any kind of programming language can be designed through a parser-interpreter. This means, thinking vice-versa, one parser-interpreter can be rewritten to any kind of syntaxed-programming language.

Also, there is no need to use the compiler if we use interpreter, (like Java), so it produces the result directly.

This sounds more like Utopia, but if there is SINGLE MOST POWERFUL language that can do everything perfectly, we would not need to implement a programming language as parser-interpreter architecture at all.

2. Based on the following grammar, WAE, write an expression that contains two binding identifiers, three bound identifiers, two free identifiers. By considering the scope of each binding identifier and inner 'with' expressions, a free identifier can also be a bound identifier. For that case, you must count them as both a free identifier and a bound identifier. After writing your expression, explain or mark what are binding/bound/free identifiers in your expression. (2 points, no partial point, your expression must follow the grammar, and the designated number of binding/bound/free identifiers must be exact.)

BNF
<WAE> ::= <num> | {+ <WAE> <WAE>} | {- <WAE> <WAE>} | {with {<id> <WAE>} <WAE>} | <id>

Free = F, Bound = Bo, Binding = Bi, Free+Bound = FB

{ with { x { + 1 2 } } { with { y { - 4 3 } } { + x y { with { w x } x z } } }

BI BI BO BO BI BO FB F

First X = BI
First Y = BI
Second X = BO
Second Y = BO
First W = BI
Third X = BO
Fourth X = FB
First Z = F
Total: 3 Binding, 4 Bound, 2 Free Identifiers!

3. Write your expression in your preferred programming such as C, Java, python, etc. Although the WAE language is different from existing languages, you can make similar code by your preferred grammar. To mimic an inner 'with' expression in your prefered language, you can use a block or a function block. Also mark what are binding/bound/free identifiers in your code. (2 points)

{ with { x { + 1 2 } } { with { y { - 4 3 } } { + x y { with { w x } x z } } }

```
int theExpression(x, y)
{
        int answer = x + y;
        int w = x;
        answer = answer + x + z;
        return answer;
}

int main()
{
        Int x = 1 + 2;
        Int y = 4 - 3;
        Int answer = theExpression(x, y);
}
```