# **Quiz #5 - Programming Language**, 2017, Term 2

Name (  )  St. Id (  )

NB) 배점동일, 부분점수 인정

**1. Fill in the blanks.**

A. Defines the meaning of a program

- *(Syntactically) valid*
- *Static (type) checking valid*

B. The (*denotational) semantics* of a language defines the meanings of abstract language elements as a collection of state-transforming functions.

C. A *semantic (domain)* is a set of values whose properties and operations are independently well-understood and upon which the rules that define the semantics of a language can be based.

**2. Fill in the blanks.**

A. The meaning of a *(Program)* is defined to be the meaning of the *body* when given an initial state consisting of the variables of the *decpart* initialized to the *undef* value corresponding to the variable's type.

B. The meaning of a *(Skip)* is an identity function on the state; that is, the state is unchanged.

C. The output state is computed from the input state by replacing the value of the *(target)* variable by the computed value of the *(source)* expression.

D. The meaning of a conditional is:

   *If the test is (true), the meaning of the thenbranch;*

   *Otherwise, the meaning of the elsebranch*

E. A (*side) (effect)* occurs during the evaluation of an expression if, in addition to returning a value, the expression alters the state of the program.

F. The meaning of a Program is the meaning of its (body) when given an empty initial state.

   - *Variables declared as encountered*
   - *Type of a variable is type of is value*

G. The meaning of an expression in the current state is a value defied as follows:

   – *If the expression is a value, then the value itself*

- *If the expression is a Variable:*
    - If the Variable occurs in the current state, then its associated value.
    - Otherwise the program is (meaningless)

**3. Argument와 Parameter의 차이를 설명하시오.**

- An *argument* is an expression that appears in a function call.
- A *parameter* is an identifier that appears in a function declaration.

**4. Pass by value, Pass by reference 그리고 Pass by value-result에 대해 설명하시오.**

Pass by value : Compute the *value* of the argument at the time of the call and assign that value to the parameter. So passing by value doesn't normally allow the called function to modify an argument's value

Pass by reference : Compute the *address* of the argument at the time of the call and assign it to the parameter.

Pass by value-result : Pass by value at the time of the call and/or copy the result back to the argument at the end of the call.

- *E.g., Ada's in out parameter can be implemented as value-result.*
- *Value-result is often called* copy-in-copy-out.

Reference and value-result are the same, except when *aliasing* occurs. That is, when:

- *the same variable is both passed and globally referenced from the called function, or*
- *the same variable is passed for two different parameters.*

## 5. Activation records와 Run time stack에 대해 설명하시오.

Activation Records : lock of information associated with each function call, which includes:

- *parameters and local variables*
- *Return address*
- *Saved registers*
- *Temporary variables*
- *Return value*
- *Static link - to the function's static parent*
- *Dynamic link - to the activation record of the caller*

Run Time Stack : stack of activation records.

- Each new call pushes an activation record, and

each completing call pops the topmost one.

- So, the topmost record is the most recent call, and

the stack has all active calls at any run-time moment.

## 6. 변수 a가 값으로 *undef*를 가질 때 b = a와 같은 저장문을 허용하지 않음으로써 얻어지는 장점과 단점을 설명해 보시오.

An advantage is that undef can be explicitly tested as a value in the program

(e.g., to see if a variable has been initialized). A disadvantage is that

programs operating with undef values can propagate some very obscure

run-time errors before the program crashes or delivers incorrect results.

A whole new "arithmetic" with undefined values would have to be understood

by programmers and language designers alike in order to properly

introduce undef into the semantic domain of a programming language.

Readers may note that the current Clite implementation permits such an

Assignment but does not permit the value undef to be used in a Binary

or Unary expression. The simplest way to prevent undef from being assigned is for the semantic rule for Assignement to disallow an undef source expression.

**7. 다음 C 계열 프로그램을 생각해 보자.**

```
void swap(int[] list, int i, int j) {
        int temp = list[i];
        list[i] = list[j];
        list[j] = temp;
}

void main() {
        int x[3] = {5, 2, 4}
        swap(x, 1, 2);
}
```

다음 매개변수 전달 방법을 가정하였을 때 배열 x의 최종값은 무엇인가?

(a) 인수 x를 값 전달 방식으로 전달함 (Pass by value)

(b) 인수 x를 주소 전달 방식으로 전달함 (Pass by refrence)

(c) 인수 x를 값-결과 전달 방식으로 전달함 (Pass by value-result)

(a) When passed by value, x remains {5, 2, 4}.

(b) When passed by reference, x becomes {2, 5, 4}.

(c) When passed by value-result, x becomes {2, 5, 4}.

**<End of the Quiz>**