HCMC University of Technology
Faculty of Computer Science & Engineering




# Assignment 2

## Static Checker

cuu duong than cong . com

Author Dr. Nguyen Hua Phung

cuu duong than cong . com

October 4, 2015

CuuDuongThanCong.com https://fb.com/tailieudientucntt

# Contents

**1 Specification** **2**

**2 Static Checker** **2**
2.1 Redeclared Variable/Constant/Attribute/Class/Method/Parameter: . . . . 3
2.2 Undeclared Identifier/Attribute/Method/Class: . . . . . 3
2.3 Cannot Assign To Constant: . . . . . 3
2.4 Type Mismatch In Statement: . . . . . 3
2.5 Type Mismatch In Expression: . . . . . 4
2.6 Type Mismatch In Constant: . . . . . 4
2.7 Cannot access private attribute: . . . . . 4
2.8 Method not return: . . . . . 5
2.9 Break/Continue not in loop: . . . . . 5
2.10 Not Constant Expression: . . . . . 5

**3 Submissions** **5**
3.1 Phase 1: . . . . . 5
3.2 Phase 2: . . . . . 5
3.3 Phase 3: . . . . . 5

**4 Plagiarism** **6**

cuu duong than cong . com

cuu duong than cong . com



CuuDuongThanCong.com https://fb.com/tailieudientucntt

# Assignment 2

# version 1.0

After completing this assignment, you will be able to

- explain the principles how a compiler can check some semantic constraints such as type compatibility, scope constraints,... and
- write a medium ( 300-500LOC) Scala program to implement that.

## 1 Specification

In this assignment, you are required to write a static checker for a program written in BKOOL. To complete this assignment, you need to:

- read carefully the specification of BKOOL language
- Download and unzip file **initial.zip**, which contains the following folders:
  - src: contains bkool/Main.scala, bkool/checker/StaticError.scala, bkool/checker/TestChecker.scala, bkool/checker/StaticCheck.scala.
  - checkertestcases: contains some sample inputs
  - checkersolutions: contains output of the checker.
- Modify StaticCheck.scala to implement the static checker. **Please fill in your id in the headers of this file**.

## 2 Static Checker

A static checker plays an important role in modern compilers. It checks in the compiling time if a program conforms to the semantic constraints according to the language specification. In this assignment, you are required to implement a static checker for BKOOL language. The input of the checker is in the AST of a BKOOL program, i.e. the output of the assignment 1. The output of the checker is nothing if the checked input is correct, otherwise, an error message is released and the static checker will stop immediately

For each semantics error, students should throw corresponding exception given in StaticError.scala to make sure that it will be printed out the same as expected. Every test-case has at most one kind of error. The semantics constraints required to check in this assignment are as follows.



CuuDuongThanCong.com https://fb.com/tailieudientucntt

### 2.1 Redeclared Variable/Constant/Attribute/Class/Method/Parameter:

An identifier must be declared before used. However, the declaration must be unique in its scope. Otherwise, the error messages "Redeclared <kind>: "+<identifier> is released, where <kind> is the kind of the identifier in the second declaration. The scope of an identifier (variable, constant, attribute, class, method, parameter) is informally described as in Section 7 of BKOOL specification.

Note that an attribute/method of a class is redeclared if there is another previous member of the same class with the same name.

### 2.2 Undeclared Identifier/Attribute/Method/Class:

The error message "Undeclared Identifier: "+<identifier> is released when there is an identifier is used but its declaration cannot be found. The identifier can be a variable, constant or parameter. "Undeclared Class: "+<class name> is released in similar situation for a class usage. "Undeclared Attribute: "+<attribute name> is released when there is an access to an attribute of a class but there is no declaration of the attribute of the class. "Undeclared Method: "+<method name> is released in similar situation for a method invocation.

Note that BKOOL is an OO language so all members of a class may be inherited by its subclasses.

### 2.3 Cannot Assign To Constant:

cuu duong than cong . com

The left-hand side (LHS) of an assignment statement operator cannot be declared as a constant, otherwise, the error message "Cannot Assign To Constant: "+<statement> is released.

An assignment operator may appear in an assignment or a for statement. If the error happens in an assignment statement, the assignment statement is passed in the error message. In the case of a for statement, just the assignment part in this statement is printed out in the error message.

### 2.4 Type Mismatch In Statement:

A statement must conform the corresponding type rules for statements, otherwise the error message "Type Mismatch In Statement: "+<statement> is released.

The type rules for statements are as follows:

- The type of a conditional expression in an **if**, or **while** statement must be boolean.

- For an assignment statement, the left-hand side can be in any type except void type. The right- hand side (RHS) is either in the same type as that of the LHS or in the type that can coerce to the LHS type. In BKOOL, just the integer can coerce to the float or a subtype can coerce to its super type. When LHS is in an array type, RHS



CuuDuongThanCong.com https://fb.com/tailieudientucntt

must be in array type whose size is the same and whose element type can be either the same or able to coerce to the element type of LHS.

- For a call statement E.<method name>(<args>), E must be in class type; the callee must have void as return type. In addition, for parameter passing, the rule for an assignment is applied to parameter passing where a parameter is considered as the LHS and the corresponding argument is the RHS.

- For a return statement, the return expression can be considered as RHS of an implicit assignment whose LHS is the return type.

### 2.5 Type Mismatch In Expression:

An expression must conform the type rules for expressions, otherwise the error message "Type Mismatch In Expression: "+<expression> is released.

The type rules for expression are as follows:

- For an array subcripting E1[E2], E1 must be in array type and E2 must be integer.

- For a binary and unary expression, the type rules are described in the BKOOL specification.

- For a method call E.<method name>(<args>), E must be in class type; the callee <method name> must have non-void as return type. The type rules for arguments and parameters are the same as those mentioned in a procedure call.

- For an attribute access E.id, E must be in class type.

### 2.6 Type Mismatch In Constant:

The types of the left and right hand sides of a constant declaration must conform the type rule for an assignment described above. Otherwise, the error message "Type Mismatch In Constant: "+<constant declaration> is released.

The following constraints are requires just for **gifted students (KSTN)**

### 2.7 Cannot access private attribute:

As mentioned in Section 7 of BKOOL specification, all attributes of a class are visible only in the class and its subclasses so an access to an attribute of another class that is not a superclass of the current class will cause this error. The error message "Cannot Access Private Attribute: <field name> of class <class name>" will be released in such case.



CuuDuongThanCong.com https://fb.com/tailieudientucntt

### 2.8 Method not return:

A method that does not return **void** must return something in every its execution paths. If there exists one path where there is nothing returned, the error message "Method Not Return: <method name>" will be released.

### 2.9 Break/Continue not in loop:

A break/continue statement must be inside directly or indirectly a loop otherwise the error message "Break/Continue Not In Loop: <line>" will be released.

### 2.10 Not Constant Expression:

The expression to initialize a constant must be evaluated statically so its operands must be a literal or an immutable attribute and they are combined together by operators only. If this constraint is violated, the error message "Not Constant Expression: <the violating subexpression>" is released.

## 3 Submissions

The operating system when cheking your submission is Windows. Make sure that your program can be compiled and run in the environment. The Scala compiler version is 2.11.2.

There are 3 phases to submit, and there is only one file to submit in all 3 phases: StaticCheck.scala with the entry method **test**.

### 3.1 Phase 1:

- The deadline is **Friday, October 16th, 2015 at 16:30**.
- The constraints required in this phase are: Redeclared Class/Method/Attribute.
- There will be around 50 test-cases for this phase.

### 3.2 Phase 2:

- The deadline is **Friday, October 23rd, 2015 at 16:30**.
- The constraints required in this phase are: Redeclared Parameter/Variable/Constant and Undeclared Identifier/Class/Method/Attribute.
- There will be around 100 test-cases for this phase.



CuuDuongThanCong.com https://fb.com/tailieudientucntt

### 3.3 Phase 3:

- The deadline is **Friday, October 30th, 2015 at 16:30**.
- All required constraints.
- There will be around 150 test-cases (+ 100 test-cases for KSTN) for this phase.

## 4 Plagiarism

- You must complete the assignment by yourself and do not let your work seen by someone else.
- You just submit your code in your allocated account.

If you violate any requirement, you will be punished by the university rule for plagiarism.

cuu duong than cong . com

cuu duong than cong . com



CuuDuongThanCong.com https://fb.com/tailieudientucntt