Handout Phase 1

Interpretation and Compilation
30-SET-2019
due week **11-NOV-2019**

Luis Caires

Goal

Implement a complete interpreter for the basic imperative-functional language specified

Use the approach developed in the lectures

- LL(1) parser using JAVACC
- AST model
- Environment based evaluator
- Dynamic type checking – issue proper error messages for runtime type errors

Fully understanding the handout statement is part of the handout as well. Contact me if you need help.

Submission Instructions

Create a bitbucket repository

Add me (lcaires@fct.unl.pt) as a team member

Send me the repository URL in an email with subject

ICL HO1 XXXXX YYYYY

where XXXXX etc are the student numbers (members of the group)

Abstract Syntax

EE -> EE **;** EE | EE **:=** EE

| **num** | **id** | **bool** | **let** (**id** = EE)+ **in** EE **end**

| **fun id***-> EE **end**

| EE **(** EE* **)**

| **new** EE | **<!>** EE

| **if** EE **then** EE **else** EE **end**

| **while** EE **do** EE **end**

| EE **binop** EE

| **unop** EE

Concrete Syntax

EM -> E(**<;>**EM)*	ASTSeq(E1,E2)
E -> EA(**< == >** EA)?	ASTEq(EA,EA)
EA -> T(**<+>**EA)*	ASTAdd(E1,E2)
T -> F ((<*>T)*	ASTMul(F,T)
| (**<(>**AL**<)>**)*	ASTApply(F,AL)
| **<:=>** E)	ASTAssign(F,E)

AL -> (EM(**<,>**EM)*)?

PL -> (id(**<,>**id)*)?

F -> **num** | **id** | **bool** | **let** (**id** = EM)+ **in** EM **end**

| **fun** PL -> EM **end** | **<(>** EM **<)>**

| **new** F | **<!>** F

| **if EM then EM else EM end** ASTIf(EM,EM,EM)

| **while EM do EM end** ASTWhile(EM,EM)

Basic operations

Arithmetic operations (on integer values)

E+E, E-E, E*E, E/E, -E

Relational operations

E==E, E>E, E<E, E<=E, E>=E

Logical operations (on boolean values)

E && E, E || E, ~E

FIRST PHASE Handout

Implement a complete interpreter for the language

Use the approach developed in the lectures

- LL(1) parser using JAVACC
- AST Model
- Interpreter
- Compiler

Fully understanding the handout statement is part of the handout as well. Contact me if you need help.

DUE Week of 11 Nov 2019

FIRST PHASE Handout

Abstract Syntax

EE ->

| **num** | **id**

| EE **+** EE | EE **-** EE

| EE ***** EE | EE **/** EE | **-**EE | **(** EE **)**

| **let** (**id** = EM)+ **in** EM **end**