Exemples de programmes LOOC

Suzanne Collin, Sébastien Da Silva, Pierre Monnin

14/01/2017

1 Introduction

Ce document a pour objectif de fournir un guide de progression dans le support des structures du langage LOOC lors de l'étape de génération de code. Les codes présentés ne sont que des exemples et ne couvrent pas forcément l'ensemble des structures du niveau associé.

Les entrées / sorties (`read` et `write`) peuvent être implémentées en parallèle de n'importe quel niveau. Il est préférable de les avoir le plus tôt possible afin de pouvoir aisément montrer des résultats en soutenance. La récursivité peut être supportée à partir de n'importe quel niveau concernant les fonctions.

2 Niveau 1 : Programme principal, variables et structures de contrôle

- Exécution du programme principal
- Définition de variables dans le programme principal
- Structures de contrôle : `if`, `for`

```
var n : int;
var total : int;
var i : int;

n := 10;
total := 0;

for i in 0 .. n
do
  total := total + i * i;
end

write total;
```

3 Niveau 2 : Appels de fonctions

- Classes avec méthodes et sans attributs
- Instanciation des classes
- Appels des méthodes

```
class Math =
(
  method pow(a : int, b : int) : int
  {
    var retval : int;
    var i : int;

    if b == 0
    then
```

```
      return (1);
    fi

    retval := 1;
    for i in 1 .. b
    do
      retval := retval * a;
    end

    return (retval);
  }
)

var m : Math;
m := new Math;
write m.pow(2, 3);
```

4 Niveau 3 : Objets avec attributs

— Classes définies avec des attributs et des méthodes
— Modification des attributs

```
class Fibonacci =
(
  var current : int;
  var previous : int;
  var generation : int;

  method init()
  {
    current := 1;
    previous := 0;
    generation := 1;
  }

  method nextGen()
  {
    var temp : int;
    temp := current;
    current := current + previous;
    previous := temp;
    generation := generation + 1;
  }

  method nextGenAndGet() : int
  {
    do this.nextGen();
    return (current);
  }
)

var f : Fibonacci;
f := new Fibonacci;
do f.init();
do f.nextGen();
do f.nextGen();
write f.nextGenAndGet();
```

5 Niveau 4 : Héritage

— Héritage

— Liaison statique dans les classes
— Liaison dynamique à l'appel des méthodes

```
class Animal =
(
  var name : string;

  method setName(n : string)
  {
    name := n;
  }

  method whoami()
  {
    write "My name is ";
    write name;
  }
)

class Dog inherit Animal =
(
  method whoami()
  {
    write "My name is ";
    write name;
    write ". And I am a dog.";
  }
)

class Cat inherit Animal =
(
  method whoami()
  {
    do super.whoami();
    write ". And I am a cat.";
  }
)

var a1 : Animal;
var a2 : Animal;

a1 := new Dog;
do a1.setName("boo");
a2 := new Cat;
do a2.setName("foo");

do a1.whoami();
do a2.whoami();
```