ENGF0002 (Design and Professional Skills)

Scenarios

The focus of this document is on the differences in the implementation of mutable structures between the three prototypes of the programming language. It follows an observation-explanation-conclusion structure in that an observation is laid down, explained, and a theory is formulated it out of it.

```
#lang MutableStructures

record = {"a": 7, "b": 26, "c": 19 }

fun f(map):
  map["b"] := 11
  print(map["b"])
 end

f(record)
print(record["b"])
```

```
Welcome to DrRacket, version 7.0 [3m].
Language: MutableStructures, with debugging; memory limit: 128 MB.
version: 2018-09-04T22:54:09-04:00

------Core 1------

11
11

------Core 2------

11
26

------Advanced 1------

11
26
```

Partition after test: {Core-1, Core-3}, {Core-2}

Classifier-1 (1.rkt)

Observation-1: When a field is changed in a function, then Core-1 keeps the change in the record, while Core-2 and Core-3 do not keep the change.

Theory-1: This indicates that a mutable structure passed as a parameter to a function in Core-1 is passed by reference. Therefore, any changes made to the parameter are kept. Core 2 and Core 3, on the other hand, pass a copy of the record to the function and therefore any changes made are local to that function.

Classifier-2 (2.rkt)

```
#lang MutableStructures

record = {"x" : 7}

fun f(map):
  map["x"] := map
  print(map)
end

f(record)
```

```
Welcome to DrRacket, version 7.0 [3m].
Language: MutableStructures, with debugging; memory limit: 128 MB.
version: 2018-09-04T22:54:09-04:00

------Core 1------

{"x":{"x":{"x":{"x":...}}}}

------Core 2------

ERROR: Program timed out


------Advanced 1------

{"x":{"x":{"x":{"x":...}}}}
```

Observation-2: When an attempt is made to assign a record to a field of the same record, then Core-1 and Core-2 both perform the operation multiple times (until a pre-defined limit is reached) Core-3, on the other hand, times out.

Theory-2: *Recursion occurs when a thing is defined in terms of itself or of its type.*

This implies that when a record is added to a single field of the same record, then Core-1 and Core-2 perform this recursive operation until a pre-defined recursion limit is reached. When the record is printed, then both prototypes return a text representation of the record, where the ellipses represent the '*n*' number of records added to the field.

Core-3, on the other hand, returns an error and does not compile after its recursion limit is reached.

Partition after test: {Core-1, Core-3}, {Core-2}