

CS320 Programming Languages

Homework #7: KXCFAE

Due: 13 November 2019

The goal of Homework #7 is to implement the interpreter of **KXCFAE**, which is the extended version of **KCFAE** with multiple arguments and **eXceptions**.

1 Multiple Arguments

Implement the interpreter to support multiple or zero arguments to a function, and multiple or zero arguments in a function call:

```
<KXCFAE> ::= ...
           | {fun {<id>*} <KXCFAE>}
           | {<KXCFAE> <KXCFAE>*}
```

As usual, your interpreter should detect the mismatch between the numbers of function parameter and arguments at function calls and report an error that includes the words **"wrong arity"**. Note that, the function **run** takes an string, parses it, interprets it with an empty substitution, and produces a number string, a string **"function"** for closures or a string **"continuation"** for continuations.

Assume that each argument **<id>** is distinct for a **fun** expression. All continuations still accept a single argument. Example:

```
test(run("{fun {x y} {- y x}} 10 12}"), "2")
test(run("{fun {} 12}"), "function")
testExc(run("{fun {x y} 1} 2}"), "wrong arity")
test(run("{withcc esc {fun {x y} x} 1 {esc 3}}}"), "3")
```

2 Exceptions

Extend your language to support catching and throwing exceptions:

```
<KXCFAE> ::= ...
           | {try <KXCFAE> catch <KXCFAE>}
           | {throw}
```

The **try-catch** form evaluates its first **<KXCFAE>** and returns its value - unless evaluating the first **<KXCFAE>** throws an exception by evaluating **throw**, in which case the result of the **try** expression is the result of the second **<KXCFAE>**. Throw an error with the words **"no enclosing try-catch"** when **throw** is used without a dynamically enclosing **try-catch** from.

```
test(run("{try 1 catch 2}"), "1")
test(run("{try {throw} catch 2}"), "2")
test(run("{try {+ 1 {throw}} catch 2}"), "2")
test(run("{fun {f} {try {f} catch 1}} {fun {} {throw}}}"), "1")
testExc(run("{throw}"), "no enclosing try-catch")
```