Programming Paradigms Final Project: Test Report

Group 33
Martijn Verkleij & Wouter Bos

University of Twente
m.f.verkleij@student.utwente.nl, w.f.a.bos@student.utwente.nl
s1466895 s1606824

July 7, 2017

Contents

Chapter 1

Syntax Tests

1.1 Syntax 1

1.1.1 Source

```
int number;

procedure p() {}
```

1.1.2 Output

```
Main: tokenList not fully parsed
```

1.2 Syntax 2

1.2.1 Source

```
procedure(j, int i) {};
```

1.2.2 Output

```
Main: tokenList not fully parsed
```

1.3 Syntax 3

1.3.1 Source

```
int 5num = 5;
```

1.3.2 Output

```
Main: tokenList not fully parsed
```

1.4 Syntax 4

1.4.1 Source

```
if (true) else {// do nothing}
```

1.4.2 Output

```
Main: tokenList not fully parsed
```

1.5 Syntax 5

1.5.1 Source

```
print(5*(3&&||/1));
```

1.5.2 Output

```
Main: tokenList not fully parsed
```

Chapter 2

Contextual Tests

2.1 Wrong Type

2.1.1 Source

```
int i = 1;
if (i) {
    print(i);
}
```

2.1.2 Output

```
Main: Condition in if statement should be of type: bool, but isnt, in: ASTVar "i" ([],[],[],[])
```

2.2 Not Declared

2.2.1 Source

```
if (i) {
    print(1);
}
```

2.2.2 Output

```
Main: Variable: i not declared in Checker.getExprType.iterVar
```

Chapter 3

Semantic Tests

3.1 Banking

This test has been run on 3 Sprockells.

3.1.1 Source

```
global int john = 10000;
global int jane = 2000;
global int martijn = 99999;

procedure deposit(int account, int amount) {
    account = (account + amount);
}

procedure withdraw(int account, int amount) {
    if ((account >= amount)) {
        account = (account - amount);
    }
}

procedure transfer(int sender, int target, int amount) {
    if ((sender >= amount)) {
        sender = (sender - amount);
        target = (target + amount);
    }
}

procedure test1() {
    print(john, jane, martijn);
    fork deposit(jane, 100);
    fork deposit(john, 100);
    fork deposit(martijn, 1);
    join;
    print(john, jane, martijn);

    fork deposit(jane, 200);
    fork withdraw(john, 200);
    fork deposit(martijn, 2000);
    join;
    print(john, jane, martijn);
    fork withdraw(jane, 10);
    fork withdraw(john, 20);
    fork deposit(martijn, 100);
    join;
    print(john, jane, martijn);
    fork withdraw(jane, 300);
    fork withdraw(john, 30000);
    fork withdraw(martijn, 50);
    join;
    print(john, jane, martijn);
    fork withdraw(jane, 35);
    fork transfer(martijn, john, 1000);
    join;
    print(john, jane, martijn);
    fork transfer(martijn, jane, 100);
    join;
    print(john, jane, martijn);

}

test1();
```

3.1.2 Results

```
Sprockell 0 says 10000
Sprockell 0 says 2000
Sprockell 0 says 99999
Sprockell 0 says 10100
Sprockell 0 says 2100
Sprockell 0 says 100000
Sprockell 0 says 9900
Sprockell 0 says 2300
Sprockell 0 says 102000
Sprockell 0 says 9880
Sprockell 0 says 2290
Sprockell 0 says 102100
Sprockell 0 says 9880
Sprockell 0 says 1990
Sprockell 0 says 102050
Sprockell 0 says 10880
Sprockell 0 says 1955
Sprockell 0 says 101050
Sprockell 0 says 10880
Sprockell 0 says 2055
Sprockell 0 says 100950
```

3.2 Blocks

3.2.1 Source

```
int x = 1;
int y = 100;
bool a = false;
bool b = false;
print(x,y,a,b);
{
    int x = 2;
    int y = 120;
    bool a = true;
    bool b = false;
    print(x,y,a,b);
    {   {}
        int x = 3;
        int y = 123;
        bool a = false;
        bool b = true;
        print(x,y,a,b);
        {
            int x = 4;
            int y = 423;
            bool a = true;
            bool b = true;
            print(x,y,a,b);
        }
        print(x,y,a,b);
        {
            int x = 5;
            int y = 453;
            bool a = true;
            bool b = false;
            print(x,y,a,b);
            {
                int x = 5;
                int y = 453;
                bool a = false;
                bool b = true;
                print(x,y,a,b);
            }
            print(x,y,a,b);
            {
                int x = 6;
                int y = 456;
                bool a = false;
                bool b = false;
                print(x,y,a,b);
            }
            print(x,y,a,b);
        }
        print(x,y,a,b);
    }
    print(x,y,a,b);
    {
        print(x,y,a,b);
        {
            print(x,y,a,b);
        }
    }
    print(x,y,a,b);
}
print(x,y,a,b);
```

3.2.2 Results

```
Sprockell 0 says 1
Sprockell 0 says 100
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 2
Sprockell 0 says 120
Sprockell 0 says 1
Sprockell 0 says 0
Sprockell 0 says 3
Sprockell 0 says 123
Sprockell 0 says 0
Sprockell 0 says 1
Sprockell 0 says 4
Sprockell 0 says 423
Sprockell 0 says 1
Sprockell 0 says 1
Sprockell 0 says 3
Sprockell 0 says 123
Sprockell 0 says 0
Sprockell 0 says 1
Sprockell 0 says 5
Sprockell 0 says 453
Sprockell 0 says 1
Sprockell 0 says 0
Sprockell 0 says 5
Sprockell 0 says 453
Sprockell 0 says 0
Sprockell 0 says 1
Sprockell 0 says 5
Sprockell 0 says 453
Sprockell 0 says 1
Sprockell 0 says 0
Sprockell 0 says 6
Sprockell 0 says 456
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 5
Sprockell 0 says 453
Sprockell 0 says 1
Sprockell 0 says 0
Sprockell 0 says 3
Sprockell 0 says 123
Sprockell 0 says 0
Sprockell 0 says 1
Sprockell 0 says 2
Sprockell 0 says 120
Sprockell 0 says 1
Sprockell 0 says 0
Sprockell 0 says 2
Sprockell 0 says 120
Sprockell 0 says 1
Sprockell 0 says 0
Sprockell 0 says 2
Sprockell 0 says 120
Sprockell 0 says 1
Sprockell 0 says 0
Sprockell 0 says 2
Sprockell 0 says 120
Sprockell 0 says 1
Sprockell 0 says 0
Sprockell 0 says 1
Sprockell 0 says 100
Sprockell 0 says 0
Sprockell 0 says 0
```

3.3 Call-by-reference

This test has been run on 3 Sprockells.

3.3.1 Source

```
global int var = 1337;
global int y = 42;
```

```

procedure write(int input, int output) {
    output = input;
}

print (y);
fork write(var, y);
join;
print(y);
```

3.3.2 Results

```
Sprockell 0 says 42
Sprockell 0 says 1337
```

3.4 Cyclic Recursion

3.4.1 Source

```
procedure prod(int i) {
    i = (i + 1);
    //print (i);
    cons(i);
}

procedure cons(int i) {
    if ((i > 1)) {
        i = (i - 2);
        print(i);
        prod(i);
    } else if ((i > 0)) {
        i = (i - 1);
        print(i);
    }
}

prod(18);
```

3.4.2 Results

```
Sprockell 0 says 17
Sprockell 0 says 16
Sprockell 0 says 15
Sprockell 0 says 14
Sprockell 0 says 13
Sprockell 0 says 12
Sprockell 0 says 11
Sprockell 0 says 10
Sprockell 0 says 9
Sprockell 0 says 8
Sprockell 0 says 7
Sprockell 0 says 6
Sprockell 0 says 5
Sprockell 0 says 4
Sprockell 0 says 3
Sprockell 0 says 2
Sprockell 0 says 1
Sprockell 0 says 0
Sprockell 0 says 0
```

3.5 Deep Expression

3.5.1 Source

```
int a = 100;
//100000;
//200000;
//300000;
//400000;
a = ((a + (((10 * (-15)) * 42) * (3 + 2))) * (2 * (7 + 11 - 98))); // comment test
a = 3 * 5 + 100 - 1 - 1 + a;
print(a);
print(--a);
print(a);
```

3.5.2 Results

```
Sprockell 0 says 5024113
Sprockell 0 says 5024112
Sprockell 0 says 5024112
```

3.6 Fib

3.6.1 Source

```
procedure fib(int i, int res) {
    if ((i < (3))) {
        res = 1;
    } else {
        int a;
        int b;
        fib((i-1), a);
        fib((i-2), b);
        res = (a + b);

    }
}

int a = 0;
fib(8, a);
print(a);
```

3.6.2 Results

```
Sprockell 0 says 5024113 21
```

3.7 If

3.7.1 Source

```
bool condition = true;
if (condition) print(1); else print(0);
```

3.7.2 Results

```
Sprockell 0 says 1
```

3.8 If Else

3.8.1 Source

```
int i = 4;
print (i);
if ((i == 2)) {
    print(3,i);
} else if ((i == 1)) {{{
    print(4,i);
}}} else {{{
    print(5,i);
}}}


if ((i == 4)) {
    print(3,i);
} else if ((i == 1)) {{{
    print(4,i);
}}}
```

3.8.2 Results

```
Sprockell 0 says 4
Sprockell 0 says 5
Sprockell 0 says 4
Sprockell 0 says 3
Sprockell 0 says 4
```

3.9 Infinite Busy Loop

3.9.1 Source

```
int i = 0;
int j = 1;
while (true) {
    i = (i + j);
    j = (j * i);
    print(i,j);
}
```

3.9.2 Results

Gets stuck in an infinite loop, repeating the same output.

```
Sprockell 0 says 1
Sprockell 0 says 1
Sprockell 0 says 2
Sprockell 0 says 2
Sprockell 0 says 4
Sprockell 0 says 8
Sprockell 0 says 12
Sprockell 0 says 96
Sprockell 0 says 108
Sprockell 0 says 10368
Sprockell 0 says 10476
Sprockell 0 says 108615168
Sprockell 0 says 108625644
Sprockell 0 says 11798392572168192
Sprockell 0 says 11798392680793836
Sprockell 0 says -5570361874949185536
Sprockell 0 says -5558563482268391700
Sprockell 0 says 3671369242980155392
Sprockell 0 says -1887194239288236308
Sprockell 0 says -4483044364780175360
Sprockell 0 says -6370238604068411668
Sprockell 0 says -8730959061097906176
Sprockell 0 says 3345546408543233772
Sprockell 0 says -6745737849034768384
Sprockell 0 says -3400191440491534612
Sprockell 0 says -6096120617457680384
Sprockell 0 says 8950432015760336620
Sprockell 0 says -1019520187243692032
Sprockell 0 says 7930911828516644588
Sprockell 0 says -4809903748681826304
Sprockell 0 says 3121008079834818284
Sprockell 0 says 5865085819223539712
Sprockell 0 says 8986093899058357996
Sprockell 0 says 2740241432517279744
Sprockell 0 says -6720408742133913876
Sprockell 0 says 3246081813541552128
Sprockell 0 says -3474326928592361748
Sprockell 0 says -1859074291971129344
Sprockell 0 says -5333401220563491092
Sprockell 0 says 681350175863603200
Sprockell 0 says -4652051044699887892
Sprockell 0 says -8143132099134619648
Sprockell 0 says 5651560929875044076
Sprockell 0 says 6951259845357993984
Sprockell 0 says -5843923298476513556
Sprockell 0 says 4700992750881865728
Sprockell 0 says -1142930547594647828
Sprockell 0 says -8561800288468467712
Sprockell 0 says 8742013237646436076
Sprockell 0 says 7566188111470788608
Sprockell 0 says -2138542724592326932
Sprockell 0 says -6956372574427152384
Sprockell 0 says -9094915299019479316
Sprockell 0 says -8878846665360932864
Sprockell 0 says 472982109329139436
Sprockell 0 says 1756403854674493440
Sprockell 0 says 2229385964003632876
Sprockell 0 says -5152117973711847424
Sprockell 0 says -2922732009708214548
Sprockell 0 says 1585267068834414592
Sprockell 0 says -1337464940873799956
Sprockell 0 says 5188146770730811392
Sprockell 0 says 3850681829857011436
Sprockell 0 says 6917529027641081856
Sprockell 0 says -7678533216211458324
Sprockell 0 says -9223372036854775808
Sprockell 0 says 1544838820643317484
Sprockell 0 says 0
Sprockell 0 says 1544838820643317484
Sprockell 0 says 0
Sprockell 0 says 1544838820643317484
Sprockell 0 says 0
...
```

3.10 Infinite Empty Loop

3.10.1 Source

```
while (true) {
    // do nothing
}
```

3.10.2 Results

No output, gets stuck in an infinite loop.

3.11 Join Test

This test has been run on 2 Sprockells.

3.11.1 Source

```
global int after_a_very_long_time = 10000;

procedure ending(int j) {
    while( (j > 0) ){
        j = (j-1);

        if ( (j == 5000) ) {
            print(j);
        }
    }

}

fork ending(after_a_very_long_time);
join;
print(10000);
```

3.11.2 Results

Sprockell 1 says 5000 Sprockell 0 says 10000

3.12 Multiple Globals

This test has been run on 3 Sprockells.

3.12.1 Source

```
global int a = 8;
global int b = 9;
global int c = 10;
global int d = 11;
global int e = 12;
global int f = 13;

procedure printAll() {
    print(a,b,c,d,e,f);
}

procedure printAllBW() {
    print(f,e,d,c,b,a);
}


fork printAll();
fork printAllBW();
join;
```

3.12.2 Results

```
Sprockell 1 says 8
Sprockell 1 says 9
Sprockell 1 says 10
Sprockell 1 says 11
Sprockell 1 says 12
Sprockell 1 says 13
Sprockell 2 says 13
Sprockell 2 says 12
Sprockell 2 says 11
Sprockell 2 says 10
Sprockell 2 says 9
Sprockell 2 says 8
```

3.13 Nested Procedures

3.13.1 Source

```
procedure p0() {
    print(90);
    p1();
    print(91);
    p2();
    print(92);
    p3();
    print(93);
    p4();
    print(94);
}

```

```
procedure p1() {
    print(10);
    p2();
    print(12);
    p3();
    print(13);
    p4();
    print(14);
}

procedure p2() {
    print(20);
    p3();
    print(23);
    p4();
    print(24);
}

procedure p3() {
    print(30);
    p4();
    print(34);
}

procedure p4() {
    print(40);
}

p0();
```

3.13.2 Results

```
Sprockell 0 says 90
Sprockell 0 says 10
Sprockell 0 says 20
Sprockell 0 says 30
Sprockell 0 says 40
Sprockell 0 says 34
Sprockell 0 says 23
Sprockell 0 says 40
Sprockell 0 says 24
Sprockell 0 says 12
Sprockell 0 says 30
Sprockell 0 says 40
Sprockell 0 says 34
Sprockell 0 says 13
Sprockell 0 says 40
Sprockell 0 says 14
Sprockell 0 says 91
Sprockell 0 says 20
Sprockell 0 says 30
Sprockell 0 says 40
Sprockell 0 says 34
Sprockell 0 says 23
Sprockell 0 says 40
Sprockell 0 says 24
Sprockell 0 says 92
Sprockell 0 says 30
Sprockell 0 says 40
Sprockell 0 says 34
Sprockell 0 says 93
Sprockell 0 says 40
Sprockell 0 says 94
```

3.14 Peterson

This test has been run on 3 Sprockells.

3.14.1 Source

```
global bool flag_0 = false;
global bool flag_1 = false;
global int turn = 0;
global int i = 0;

enum test = {die};

procedure p_0() {
    flag_0 = true;
    turn = 1;
    while ((flag_1 && (turn == 1))) {
        // wait
    }
    // begin critical section
    int j = 5;
    while ((j > 0)) {
        i = ++i;
        j = --j;
    }
    // end critical section
    flag_0 = false;
}

procedure p_1() {
    flag_1 = true;
    turn = 0;
    while ((flag_0 && (turn == 0))) {
        // wait
    }
    // begin critical section
    int j = 5;
    while ((j > 0)) {
        i = --i;
        j = --j;
    }
    // end critical section
    flag_1 = false;
}

procedure test1(int j) {
    while ((j > 0)) {
        fork p_0();
        fork p_1();
        join;
        print(i);

        fork p_1();
        fork p_0();
        join;
        print(i);

        j = --j;
    }
}

test1(10);
```

3.14.2 Results

```
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
Sprockell 0 says 0
```

3.15 Recursion

3.15.1 Source

```
procedure rec(int i) {
    if ((i < 3)) {
        print(i);
        i = (i + 1);
        rec(i);
    } else {
        print(i);
    }
}

int i = 0;
rec(i);
```

3.15.2 Results

```
Sprockell 0 says 0
Sprockell 0 says 1
Sprockell 0 says 2
Sprockell 0 says 3
```

3.16 Simple Concurrency

This test has been run on 3 Sprockells.

3.16.1 Source

```
global int num = 5;

procedure set_four() {
    (3+(2*(2*(2*332))));
    num = 4;
    print(num);
}

procedure set_six() {
    num = 6;
    print(num);
}
```

```

fork set_four();
fork set_six();
join;
print(num);
```

3.16.2 Results

```
Sprockell 1 says 4
Sprockell 2 says 6
Sprockell 0 says 6
```

3.17 Simple Procedures

3.17.1 Source

```
global int a = 3;

procedure p0(int c) {
    a = c;
    c = (c + 2);
}

int b = 2;
print(b); // should print 2

a = 1;

p0(b);
print(b); // should print 4
print(0);
print(a); // should print 2
```

3.17.2 Results

```
Sprockell 0 says 2
Sprockell 0 says 4
Sprockell 0 says 0
Sprockell 0 says 2
```

3.18 While

3.18.1 Source

```
int i = 100;
while ((i >= 0)) {
    1;
    print(i);
    i = (i - 1);
}
```

3.18.2 Results

```
Sprockell 0 says 100
Sprockell 0 says 99
Sprockell 0 says 98
Sprockell 0 says 97
Sprockell 0 says 96
Sprockell 0 says 95
Sprockell 0 says 94
Sprockell 0 says 93
Sprockell 0 says 92
Sprockell 0 says 91
Sprockell 0 says 90
...
Sprockell 0 says 10
Sprockell 0 says 9
Sprockell 0 says 8
Sprockell 0 says 7
Sprockell 0 says 6
Sprockell 0 says 5
Sprockell 0 says 4
Sprockell 0 says 3
Sprockell 0 says 2
Sprockell 0 says 1
Sprockell 0 says 0
```

3.19 Enumerations

3.19.1 Source

```
global enum boe = test;
global enum boe2 = test;
global enum boe3 = bar;

enum bla = {test, test1, test2};
enum foo = {bar,baz};

procedure prok() {
    boe = test2;
}

bool b = false;

bool a = (true);
print(a);
```

```

a = (test == bar);

print(a);
a = (test == test);
print(a);

//test + test;
//test - test;

//test * test;
//test < test;
//test > test;

print(test != test);
//test && test;
//test || test;
//test <> test;
//test <= test;
//test >= test;
```

3.19.2 Results

```
Sprockell 0 says 1
Sprockell 0 says 0
Sprockell 0 says 1
Sprockell 0 says 0
```