Testing Techniques

Assignment 1

Sasja Gillissen, Martin Huijben, Martijn Terpstra

September 27, 2016

1 Introduction

2 Test Goal

3 The Product

4 The Specification

the program will continuously prompt the user for input. It will continue to prompt the user for input until it receives the exit function as input and then terminates.

Depending on the input given the program will either

- Evaluate an expression
- Assign a value to a variable and store it in memory
- Assign a value to a function and store it in memory

The full specification is included in appendixA

5 Risks

6 Test Environment

7 Quality Characteristics

8 Levels and Types of Testing

9 Who will do the Testing

Test will be performed by independent testers. This is because little knowledge is needed to preform the test and those already familiar with the

program may be biased.

10 Test Generation Techniques

11 Test Automation

Tests will be done manually. Each test is described in a spreadsheet.

To start a test, a new spreadsheet is copied from a template.

This template contains:

- One or more lines of input to be entered in the program
- A description of the expected output
- A field to insert the resulting output after entering the input.

The template is updated if and only if the tests changed.

If the resulting output is equal to the expected output, the test is successful. If not the test is unsuccessful.

The results of the test are recorded in the the spreadsheet. A new spreadsheet is created for each testing session.

After the testing session is over, the resulting spreadsheet shall be send via email to the developer team for review.

12 Exit Criteria

13 Testware

As described in section 11, the results, with the description of the test are written to a spreadsheet. The resulting spreadsheet is the test product and shall be stored in a digital format.

14 Issue Registration

After a testing as described in section 11 the developer team receives an email with a spreadsheet attached.

This spreadsheet will inform the developer if any tests have failed and if so what tests have failed and what happened when they failed.

The developer then can make bug reports based on unsuccessful test and fix them in the future.

A Specification

A.1 Overall use of the program

the program will continuously prompt the user for input. It will continue to prompt the user for input until it receives the `exit()` function as input and then terminates.

On invalid input the program the program will show an error indicating the nature of the invalid input, but will not terminate.

Valid input is on of the following

- A function assignment
- A variables assignment
- An expression
- An equality test

A.2 Memory

The program will keep defined variables and function in memory until the program terminates.

The program will be initialized with the following variables

- `pi`, whose value is `3.141592653589793`
- `e`, whose value is `2.718281828459045`

The program will be initialized with the following functions

- `floor`, which rounds down
- `ceil`, which rounds up
- `round`, which rounds to the nearest whole number.
- `log`
- `ln`, which
- `sqrt`, which returns the square root of its
- `root`, which returns the Nth root of it first argument. (WARNING: does not work properly)
- `exit` , which will terminate the program

BTW floor,ceil and round work incorrectly for NEGATIVE numbers

A.3 Expression

A valid expression is defined using Context-free Grammar

Expression → Seperator Expression Seperator
Expression → (Expression)
Expression → Number
Expression → Variable
Expression → PrefixFunction Seperator (Expression)
Expression → Expression Seperator InfixFunction Seperator Expression
Seperator → SPACE — ϕ — Seperator Seperator
Number → 0 — 1 — 2 — 3 — 4 — 5 — 6 — 7 — 8 — 9 — Number Number
Variable → Name
PrefixFunction → Name
Name → SmallLetter — CapitalLetter — NameName
SmallLetter → a—b—c—d—e—f—g—h—i—j—k—l—m—n—o—p—q—r—s—t—u—v—w—x—y—
CapticalLetter → A—B—C—D—E—F—G—H—I—J—K—L—M—N—O—P—Q—R—S—T—U—
InfixFunction → / — * — - — + — ^
variable names have a max length

A.4 Variable assignments

A variable assignment is in the form

`VARIABLENAME = EXPRESSION`

Where a valid `VARIABLENAME` is a string of 1 or more letters and is case-sensitive.

A.5 Equality tests

An equality test is in the form

`EXPRESSION1 = EXPRESSION2`

Where `EXPRESSION1` and `EXPRESSION2` are valid expressions and `EXPRESSION1` is **not** the name of a variable

A.6 Functions assignments

A function assignment is in the form

`FUNCTIONNAME(ARGUMENT):=EXPRESSION`

A.7 Expression parsing

Intermediate steps and calculations are show The result of the expression is shown

For instance an expression in the form `(1 + 2) * 4` will require the following actions.

- Evaluate `1 + 2` and print its result, (`3` in this case).
- Evaluate `3 * 4` and print its result, the `3` being the result of the previous evaluation.
- Print the results of the entire expression, `12 in this case`, after all intermediate calculations have been done.