Data Flow Testing

#### 

#### 

Team 16:

Raghav Gupta

Chelsea Jaculina

Bala Nyan Kwaw

Mike Lee

Yihua Li

**March 4th, 2018**

**Professor Badari**

**CMPE 187**

# Table of Contents

[**Table of Contents**](#_fvgztqx5fz61) **1**

[**Overview**](#_8pvx7xuczgkg) **2**

[Introduction](#_ynfxzoxduftu) 2

[Purpose](#_2shjt91jdiwk) 3

[**Part I**](#_2wvt0f6xc8n1) **4**

[Palindrome Source Code](#_dynh8zt9hdfj) 4

[**Part II**](#_12gndt7ndpng) **7**

[Test Oracle](#_jlu7nyf28ez) 7

[Environment](#_hxp4hhr1f9w7) 7

[Strategy](#_pde375v0bgqr) 7

[Test Suite](#_gxzblx6galfh) 8

[Inputs](#_x313y9bl1koy) 8

[Outputs](#_pp3ify6te8lq) 8

[Test Cases](#_gy4v0dnanl04) 9

[**Part III**](#_yjk1ob5z6wo) **13**

[Data Flow Diagram](#_59p96jyyctrz) 13

[Independent, Simple, and Loop-Free Paths](#_bjli49h8qin) 14

[P-Uses, C-Uses, and Def-Use Associations](#_uycdmzvl8ui2) 15

[**Lessons Learned**](#_zain1bfg61tw) **17**

# Overview

## **Introduction**

Through the course of completing the assigned tasks we broke it down into 3 tasks. In the first task, we were asked to write a simple (non-recursive) program to determine if a given phrase is a palindrome or not. A palindrome can be defined as a string of characters that will read the same from left to right or right to left. Our program follows a certain number of rules. One rule is that our program will check the validity for only alphanumeric characters. Character that are not alphanumeric such as spaces, tabs, or special characters (!@#s..etc), will be ignored by the program. The program considers the empty string as a valid palindrome and will also calculate the length of the phrase. For example the word “racecar” reads the same forwards as it does backwards. In addition, the program would also determine the length of the phrase.

In the second task, we were asked to write an Oracle to test the program using data flow testing. Below you will see all of our test cases in a tabular form with their respective reasons, inputs, and expected outputs. The third and last part of the homework asked us to do 5 major steps. The 5 majors steps were to identify the basic blocks in our Palindrome program and implement a flow graph. Then, we needed to identify as many independent paths as possible with a minimum of three paths. We were also asked to classify those paths as simple paths or loop free paths. The fourth and fifth step included identification. The identification of the definition of P-Uses, and C-Uses throughout the program and also the def-use associations of at least three variables.

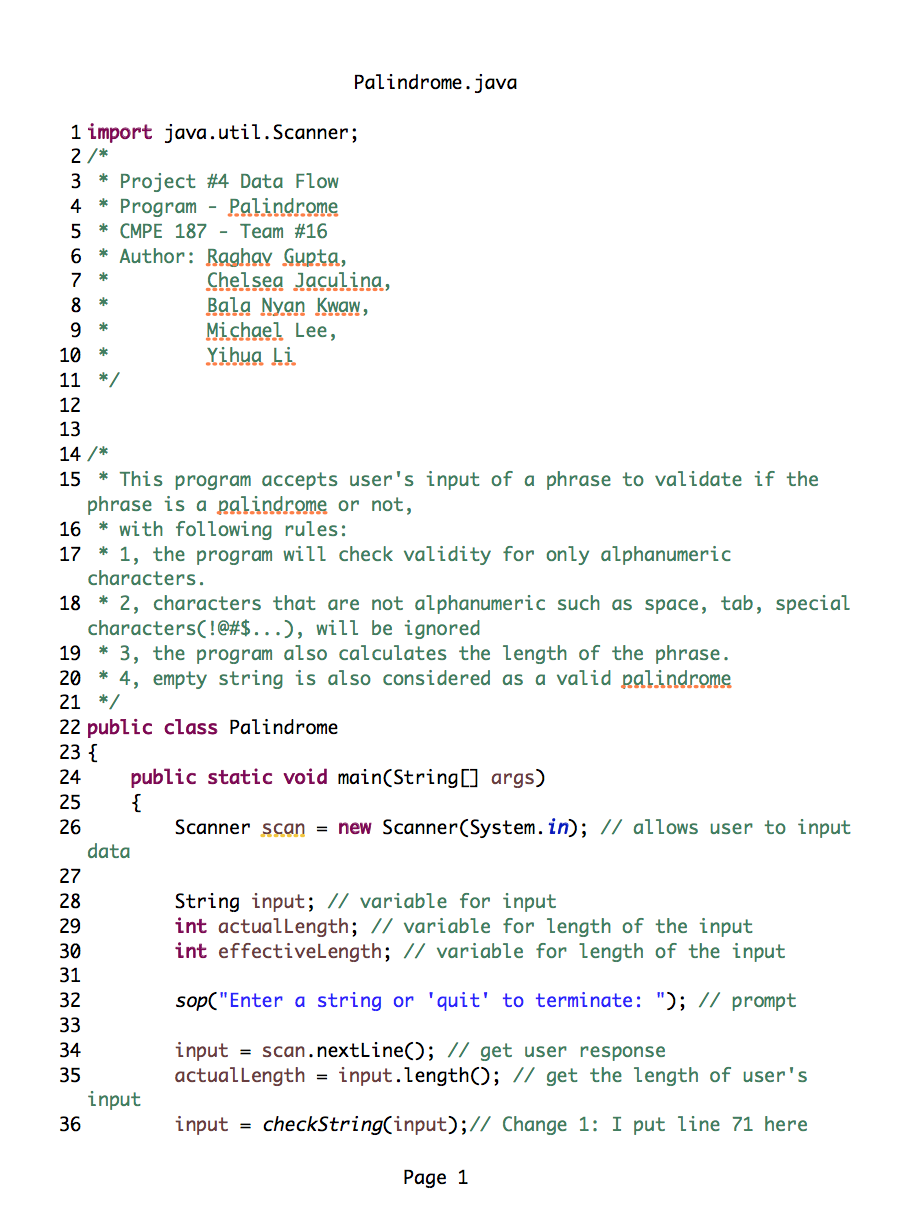
## **Purpose**

The purpose of this assignment is to use the concept of “Data Flow Testing” to test the flow of data in the previously created simple non-recursive Palindrome program that our team implemented. We understand that in order for us to master the concept of data flow testing, we needed to identify certain components so that we can generate a data flow graph. With that we, needed to identify basic blocks in our program, as many independent paths as possible with a minimum of three, identify P-uses and C-uses and their definitions, and the def-use associations of at least three variables.

In addition, this assignment allowed us to practice how to write a Test Oracle for a program by providing its test cases in a tabular form along with its respective reasons, inputs, and expected outputs.

# Part I

## Palindrome Source Code



## 

## 

## 

## 

## 

## 

# Part II

## Test Oracle

### Environment

The hardware used for programming and testing is a Macbook Pro 13.3 from 2015, with a 2.7GHz Intel Core I5 processor, 8GB 1867MHz DDR4 of memory.

The software used for programming and testing is OS X High Sierra version 10.13.3. The program is written and tested in Java code using the Eclipse version Oxygen, with Java version 8 build 1.8.0\_144. The tests are run in Eclipse IDE.

The program used for creating the Data flow diagram was the lucidchart.com (a plugin by Google) and Code2flow ( <https://code2flow.com/app> ) for verification of the diagram through a third party software.

### Strategy

The strategies that we used to test our Palindrome program was to first write our test oracle that included our inputs and expected outputs. Then, we needed to generate a data flow graph so that we were able to develop our code. From there, we were able to identify any independent paths, simple paths, or loop-free paths as well as define what P-uses and C-uses that were used throughout our program and any def-use associations of at least three variables.

## Test Suite

A test suite is a set of tests that tend be in done in units. Each test suite helps validate if the program is working the way that it is expected to. In **Table 1**, shows a table consisting of our test cases that our team has written for our Progress Bar program.

### Inputs

The inputs for our Palindrome program will be provided by a user in which they will input a string of letters, numbers, or of special characters. The user can provide phrase that contains spaces or no spaces.

### **Outputs**

The output will consist of a string telling the user whether or not the input is a palindrome or not. Below are sample outputs of our Palindrome program.

|  |  |
| --- | --- |
| If the input is a Palindrome | If the input is not a Palindrome |
| * ***input phrase*** is a Palindrome, its effective length is: ***effective length value*** , and its actual length is: ***actual length value*** | * ***input phrase*** is NOT a Palindrome, its effective length is: ***effective length value***, its actual length is: ***actual length value*** |

**Table 1:** Output Table for Palindrome Program

### Test Cases

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Purpose | Input | Expected Output | Actual Output | Pass/Fail |
| 1 | Uppercase , no space | “Racecar” | racecar is a Palindrome, its effective length is: 7, and its actual length is: 7 | Racecar is a Palindrome, its effective length is: 7, and its actual length is: 7 | Pass |
| 2 | Lowercase, no space | “racecar” | racecar is a Palindrome, its effective length is: 7, and its actual length is: 7 | racecar is a Palindrome, its effective length is: 7, and its actual length is: 7 | Pass |
| 3 | Uppercase, Two phrases | “Camp Mac” | campmac is a Palindrome, its effective length is: 7, its actual length is 8 | CampMac is a Palindrome, its effective length is: 7, and its actual length is: 8 | Pass |
| 4 | Uppercase, Multi-phrase with Spaces | “A Toyota’s a Toyota” | atoyotasatoyota is a Palindrome, its effective length is: 15, its actual length is: 18 | atoyotasatoyota is a Palindrome, its effective length is: 15, its actual length is: 18 | Pass |
| 5 | Lowercase, Multi-phrase with Spaces | “pull up if I pull up” | pullupifipullup is a Palindrome, its effective length is: 15, its actual length is: 20 | pullupifipullup is a Palindrome, its effective length is: 15, its actual length is: 20 | Pass |
| 5 | Lowercase, Two phrases with a special character | “party-trap” | partytrap is a Palindrome, its effective length is: 9, its actual length is: 10 | partytrap is a Palindrome, its effective length is: 9, its actual length is: 10 | Pass |
| 6 | Numbers | “1234321” | 1234321 is a Palindrome, its effective length is: 7, its actual length is: 7 | 1234321 is a Palindrome, its effective length is: 7, its actual length is: 7 | Pass |
| 7 | Numbers with special characters | “abc123%321cba” | abc123321cba is a Palindrome, its effective length is: 12, its actual length is: 13 | abc123321cba is a Palindrome, its effective length is: 12, its actual length is: 13 | Pass |
| 8 | Single Number | “14” | 14 is not a Palindrome, its effective length is: 2, its actual length is: 2 | 14 is not a Palindrome, its effective length is: 2, and its actual length is: 2 | Pass |
| 9 | Single Letter Uppercase | “A” | a is a Palindrome, its effective length is: 1, and its actual length is 1 | A is a Palindrome, its effective length is: 1, and its actual length is: 1 | Pass |
| 10 | Single Letter Lowercase | “a” | a is a Palindrome, its effective length is: 1, and its actual length is: 1 | a is a Palindrome, its effective length is: 1, and its actual length is: 1 | Pass |
| 10 | Single Special Character | “$” | is a Palindrome, its effective length is: 0, its actual length is: 1 | is a Palindrome, its effective length is: 0, its actual length is: 1 | Pass |
| 11 | Two Special Characters | “=)” | is a Palindrome, its effective length is: 0, its actual length is: 2 | is a Palindrome, its effective length is: 0, its actual length is: 1 | Pass |
| 12 | Multiple Special Characters | “!@#$#@!” | !@#$#@!  is a Palindrome, its effective length is: 0, and its actual length is: 7 | !@#$#@!  is a Palindrome, its effective length is: 0, and its actual length is: 7 | Pass |
| 13 | Wrong, Single-Phrase | “software” | software is NOT a Palindrome, its effective length is: 8, and its actual length is: 8 | software is NOT a Palindrome, its effective length is: 8, and its actual length is: 8 | Pass |
| 14 | Wrong, Two-Phrase | “Hello world!” | helloworld is NOT a Palindrome, its effective length is: 10, and its actual length is: 12 | helloworld is NOT a Palindrome, its effective length is: 10, and its actual length is: 12 | Pass |
| 15 | Wrong, Multi-Phrase | “Data Flow Testing” | dataflowtesting is NOT a Palindrome, its effective length is: 15, and its actual length is: 17 | dataflowtesting is NOT a Palindrome, its effective length is: 15, and its actual length is: 17 | Pass |
| 16 | Empty String | “” | is a Palindrome, its effective length is: 0, and its actual length is: 2 | is a Palindrome, its effective length is: 0, and its actual length is: 2 | Pass |
| 17 | Space | “ “ | is a Palindrome, its effective length is: 0, and its actual length is: 1 | is a Palindrome, its effective length is: 0, and its actual length is: 1 | Pass |

**Table 2:** Test Cases for Palindrome Program

# Part III

## **Data Flow Diagram**

## 

**Figure 1:** Palindrome Data Flow Diagram

## **Independent, Simple, and Loop-Free Paths**

In this section, we identify any independent paths and classify them as either a simple path or loop-free path.

* Simple path: a path in which all nodes, except possibly the first and the last, are distinct.
* Loop-free path: a path in which all nodes are distinct.

There are a total of 4 paths.

|  |  |  |
| --- | --- | --- |
| Independent Path | Simple Path? (Yes or No) | Loop-Free Path? (Yes or No) |
| 1-2-3 | Yes | Yes |
| 1-2-4-5-6-5 | Yes | No |
| 1-2-4-5-7-8 | Yes | Yes |
| 1-2-4-5-7-9 | Yes | Yes |

**Table 3:** Identification Table for Independent, Simple, or Loop-Free Paths

## P-Uses, C-Uses, and Def-Use Associations

A P-Use or Predicate-Use is defined as a variable in which its value is used to decide an execution path.

A C-Use or Computation-Use is defined as a variable in which its value is used to compute a value for defining another variable or an as output value.

A Def-Use in data flow testing is defined as a variable that gives a clear path from every definition to some P-Use or C-Use.

|  |  |  |  |
| --- | --- | --- | --- |
| Line Number | P-USE | C-USE | DEF-USE |
| 26 |  |  | scan |
| 28 |  |  | input |
| 29 |  |  | actualLength |
| 30 |  |  | effectiveLength |
| 34 |  | scan | input |
| 35 |  |  | actualLength |
| 37 |  |  | effectiveLength |
| 40 | input |  |  |
| 42 | input |  |  |
| 44 |  | input |  |
| 49 |  | input |  |
| 54 |  | scan | input |
| 55 |  | input | actualLength |
| 57 |  | input | effectiveLength |
| 67 |  | input |  |
| 71 |  |  | checkIfPalindrome |
| 73 |  |  | i |
| 74 |  |  | j |
| 76 | input, i, j |  |  |
| 79 |  | i | i |
| 80 |  | j | j |
| 82 | i, j |  |  |
| 84 |  |  | checkIfPalindrome |
| 93 |  |  | s |
| 94 |  |  | i |
| 100 | input |  |  |
| 102 |  | s, input | s |

**Table 4:** Identification Table for P-Use, C-Use, and Def-Use Associations

# 

# **Lessons Learned**

The lessons that our team learned in this project is how to write a code for a simple non-recursive Palindrome program using the high-level programming language, Java. We were able to get a grasp on how the program works by using the concept of data flow testing by identifying its independent paths, simple paths, and loop paths. Additionally, we learned what p-uses, c-uses, and def-uses are and located them in every block of our simple program.

## 

# 

## 

## 