

**MVPS’s**

**RAJARSHI SHAHU MAHARAJ POLYTECHNIC,**

**NASIK**

**SUBJECT**

**Software testing (22518)**

**MICRO-PROJECT ON**

**“Text Editor in Java”**

**Submitted By**

|  |  |  |  |
| --- | --- | --- | --- |
| **SR.**  **NO.** | **ENROLLMENT NO.** | **EXAM SEAT NO.** | **STUDENT NAME** |
| 1 | 1910020362 | 215646 | Savant Omkar Vitthal |
| 2 | 1910020363 | 215647 | Sawant Diksha Pravin |
| 3 | 1910020364 | 215648 | Shimpi Om Vilas |

**Guided By**

Prof. P. N. Patil

**COMPUTER TECHNOLOGY DEPARTMENT**

**ACADEMIC YEAR 2021-22**

# Certificate

This is to certify that Mr. /Ms. \_\_\_\_Omkar Savant\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Roll No.:- \_46\_\_ of First Semester of Diploma in Computer Technology **(CM-5-I)** of Institute **M.V.P.S’s** **RAJARSHI SHAHU MAHARAJ POLYTECHNIC, NASHIK-13, (InstituteCode-1002)** has successfully completed the Micro Project on “\_\_\_\_\_\_\_\_\_\_Text Editor\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_” in the course **Software testing (22518)** for the academic year 2021-22 as prescribed in curriculum of MSBTE, Mumbai.

|  |  |
| --- | --- |
| **Place:- Nashik** | **Enrollment No.:-** |
| **Date:-** | **Exam Seat No.:-** |

|  |  |  |
| --- | --- | --- |
| **Prof. P. N. Patil** | **Mr. P. D. Boraste** | **Dr. D. B. Uphade** |
| **Course Teacher/Guide** | **H.O.D** | **Principal** |

Seal of

Institute

**Abstract**

This paper scrutinizes the use of different practices and methods in Software Testing, enabling viewer to get the complete concept of different aspects of it. To satisfy this we have implemented various testing on text editor. A series of test cases are created on the text editor which is also used as a reference to the output, satisfying every need of a perfect microproject.

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Sr.**  **No.** | **Chapter Name** | **Page**  **No.** |
| 1 | Introduction | 5 |
| 2 | History | 7 |
| 3 | Code | 9 |
| 4 | Output | 12 |
| 5 | Testing | 17 |
| 6 | Conclusion | 22 |
| 7 | References | 23 |

**Chapter-1**

**INTRODUCTION**

1. **Software Testing:**

Software testing is the act of examining the artifacts and the behavior of the software under test by validation and verification. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but not necessarily limited to:

* analyzing the product requirements for completeness and correctness in various contexts like industry perspective, business perspective, feasibility and viability of implementation, usability, performance, security, infrastructure considerations, etc.
* reviewing the product architecture and the overall design of the product
* working with product developers on improvement in coding techniques, design patterns, tests that can be written as part of code based on various techniques like boundary conditions, etc.
* executing a program or application with the intent of examining behavior
* reviewing the deployment infrastructure and associated scripts & automation
* take part in production activities by using monitoring & observability techniques

Software testing can provide objective, independent information about the quality of software and risk of its failure to users or sponsors.



**Fig.1. Software Testing**

1. **Java:**

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers write once, run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages. As of 2019, Java was one of the most popular programming languages in use according to GitHub, particularly for client–server web applications, with a reported 9 million developers.

Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle) and released in 1995 as a core component of Sun Microsystems' Java platform. The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun had relicensed most of its Java technologies under the GPL-2.0-only license. Oracle offers its own HotSpot Java Virtual Machine, however the official reference implementation is the OpenJDK JVM which is free open-source software and used by most developers and is the default JVM for almost all Linux distributions.

A picture containing text, clipart

Description automatically generated

**Fig.2. Java**

**Chapter-2**

**History**

1. **Software Testing:**

**Glenford J. Myers** initially introduced the separation of debugging from testing in 1979. Although his attention was on breakage testing ("A successful test case is one that detects an as-yet undiscovered error."), it illustrated the desire of the software engineering community to separate fundamental development activities, such as debugging, from that of verification.

**Debugging-oriented era**: This phase was during the early 1950s, when there was no distinction between testing and debugging. The focus was on fixing bugs. Developers used to write code, and when faced with an error would analyse and debug the issues. There was no concept of testing or testers. (However, in 1957, Charles L Baker distinguished program testing from debugging in his review of the book Digital Computer Programming by Dan McCracken.)

**Demonstration-oriented era:** From 1957 to 1978, the distinction between debugging and testing was made and testing was carried out as a separate activity. During this era, the major goal of software testing was to make sure that software requirements were satisfied. As an example, the requirement might have been ‘We need a web application that displays a list of 10 products only’. Testers used to make sure that only 10 products were displayed. This failed because of the probability that a software’s function decreases as testing increases, i.e. the more you test, the more likely you'll find a bug. The concept of negative testing (or breaking the application) was not practiced in this era.



**Fig.3. Glenford J. Myers**

1. **Java:**

**James Gosling, Mike Sheridan,** and **Patrick Naughton** initiated the Java language project in June 1991. Java was originally designed for interactive television, but it was too advanced for the digital cable television industry at the time. The language was initially called Oak after an oak tree that stood outside Gosling's office. Later the project went by the name Green and was finally renamed Java, from Java coffee, a type of coffee from Indonesia. Gosling designed Java with a C/C++-style syntax that system and application programmers would find familiar.

Sun Microsystems released the first public implementation as Java 1.0 in 1996. It promised write once, run anywhere (WORA) functionality, providing no-cost run-times on popular platforms. Fairly secure and featuring configurable security, it allowed network- and file-access restrictions. Major web browsers soon incorporated the ability to run Java applets within web pages, and Java quickly became popular. The Java 1.0 compiler was re-written in Java by Arthur van Hoff to comply strictly with the Java 1.0 language specification.

In 1997, Sun Microsystems approached the ISO/IEC JTC 1 standards body and later the Ecma International to formalize Java, but it soon withdrew from the process. Java remains a de facto standard, controlled through the Java Community Process. At one time, Sun made most of its Java implementations available without charge, despite their proprietary software status. Sun generated revenue from Java through the selling of licenses for specialized products such as the Java Enterprise System.

A picture containing person, wall, holding, person

Description automatically generated

**Fig.4.**

**James Gosling, the creator of Java, in 2008**

**Chapter-3**

**Code**

import java.awt.\*;

import javax.swing.\*;

import java.io.\*;

import java.awt.event.\*;

import javax.swing.plaf.metal.\*;

import javax.swing.text.\*;

class editor extends JFrame implements ActionListener {

JTextArea t;

JFrame f;

editor()

{

f = new JFrame("editor");

try {

UIManager.setLookAndFeel("javax.swing.plaf.metal.MetalLookAndFeel");

MetalLookAndFeel.setCurrentTheme(new OceanTheme());

}

catch (Exception e) {

}

t = new JTextArea();

JMenuBar mb = new JMenuBar();

JMenu m1 = new JMenu("File");

JMenuItem mi1 = new JMenuItem("New");

JMenuItem mi2 = new JMenuItem("Open");

JMenuItem mi3 = new JMenuItem("Save");

JMenuItem mi9 = new JMenuItem("Print");

mi1.addActionListener(this);

mi2.addActionListener(this);

mi3.addActionListener(this);

mi9.addActionListener(this);

m1.add(mi1);

m1.add(mi2);

m1.add(mi3);

m1.add(mi9);

JMenu m2 = new JMenu("Edit");

JMenuItem mi4 = new JMenuItem("cut");

JMenuItem mi5 = new JMenuItem("copy");

JMenuItem mi6 = new JMenuItem("paste");

mi4.addActionListener(this);

mi5.addActionListener(this);

mi6.addActionListener(this);

m2.add(mi4);

m2.add(mi5);

m2.add(mi6);

JMenuItem mc = new JMenuItem("close");

mc.addActionListener(this);

mb.add(m1);

mb.add(m2);

mb.add(mc);

f.setJMenuBar(mb);

f.add(t);

f.setSize(500, 500);

f.show();

}

public void actionPerformed(ActionEvent e)

{

String s = e.getActionCommand();

if (s.equals("cut")) {

t.cut();

}

else if (s.equals("copy")) {

t.copy();

}

else if (s.equals("paste")) {

t.paste();

}

else if (s.equals("Save")) {

JFileChooser j = new JFileChooser("f:");

int r = j.showSaveDialog(null);

if (r == JFileChooser.APPROVE\_OPTION) {

File fi = new File(j.getSelectedFile().getAbsolutePath());

try {

FileWriter wr = new FileWriter(fi, false);

BufferedWriter w = new BufferedWriter(wr);

w.write(t.getText());

w.flush();

w.close();

}

catch (Exception evt) {

JOptionPane.showMessageDialog(f, evt.getMessage());

}

}

else

JOptionPane.showMessageDialog(f, "the user cancelled the operation");

}

else if (s.equals("Print")) {

try {

t.print();

}

catch (Exception evt) {

JOptionPane.showMessageDialog(f, evt.getMessage());

}

}

else if (s.equals("Open")) {

JFileChooser j = new JFileChooser("f:");

int r = j.showOpenDialog(null);

if (r == JFileChooser.APPROVE\_OPTION) {

File fi = new File(j.getSelectedFile().getAbsolutePath());

try {

String s1 = "", sl = "";

FileReader fr = new FileReader(fi);

BufferedReader br = new BufferedReader(fr);

sl = br.readLine();

while ((s1 = br.readLine()) != null) {

sl = sl + "\n" + s1;

}

t.setText(sl);

}

catch (Exception evt) {

JOptionPane.showMessageDialog(f, evt.getMessage());

}

}

else

JOptionPane.showMessageDialog(f, "the user cancelled the operation");

}

else if (s.equals("New")) {

t.setText("");

}

else if (s.equals("close")) {

f.setVisible(false);

}

}

public static void main(String args[])

{

editor e = new editor();

}

}

**Chapter-4**

**OUTPUT**

**Graphical user interface, application, Word

Description automatically generated**

Fig.1. Editor

**Graphical user interface, application, Word

Description automatically generated**

Fig.2. Menu

Graphical user interface, application, Word

Description automatically generated

Fig.3. Menu II

Graphical user interface, application

Description automatically generated

Fig.4.Open

Graphical user interface, text, application

Description automatically generated

Fig.5. Save

Graphical user interface, application

Description automatically generated

Fig.6.Cancel

Graphical user interface, application

Description automatically generated

Fig.7. Print

Graphical user interface, application, Word

Description automatically generated

Fig.8. Print II

Graphical user interface, application

Description automatically generated

Fig.9. Print III

Graphical user interface, application, Word

Description automatically generated

Fig.10. Cut, Copy,Paste

**Chapter-5**

**TESTING**

**MODULE – 1 (Main TextField)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case\_id** | **Test Case** | **Precondition** | **Test Data** | **Steps** | **Expected Output** | **Actual Output** | **Result** |
| 1\_1 | Test if user is able to type in the editor | Program must be running with java | "hello" | 1)Use keyboard to enter data in main field | Data must be entered and be seen | Data is entered and be seen successfully  "hello" | PASS |
| 1\_2 | Test for successful message by entering character | Program must be running with java | “hello  world” | 1)Use keyboard to enter data in main field | Data must be entered and be seen | Data is entered and be seen successfully  "hello world" | PASS |
| 1\_3 | Test for successful message by entering Number | Program must be running with java | “123456” | 1)Use keyboard to enter data in main field | Data must be entered and be seen | Data is entered and be seen successfully  "hello world" | PASS |
| 1\_4 | Test for successful message by entering  special character | Program must be running with java | “@$!%&” | 1)Use keyboard to enter data in main field | Data must be entered and be seen | Data is entered and be seen successfully  "hello world" | PASS |
| 1\_5 | Test if user is able to close program | Program must be running with java | - | 1) use close button on the program | Program must be closed | Program is closed | PASS |

**MODULE 2 – (File Menu)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case\_id** | **Test Case** | **Precondition** | **Test Data** | **Steps** | **Expected Output** | **Actual Output** | **Result** |
| 2\_1 | Test if user is able to open and view menu options | Program must be running with java | - | 1)Click on File tab on menu bar | Various Different Options must be available | Various Different Options are  available | PASS |
| 2\_2 | Test if user is able to open a file in editor | 1)Program must be running with java  2)”File” option must be clicked | - | 1)Click on File tab on menu bar.  2)Click on “Open” option | A dialog box must appear which gives user the ability to open any file from computer | A dialog box appear which gives user the ability to open file from computer | PASS |
| 2\_3 | Test if user is able to cancel operation in the editor | 1)Program must be running with java  2)”File” option must be clicked  3)”Open” or “Save” or “Print” box must be Opened | - | 1)Use keyboard to enter data in main field  2)Use Cancel button | The program must display Message  “User Canceled  the  Operation” | The program must display Message  “User Canceled  the  Operation” | PASS |
| 2\_4 | Test if user is able to use save operation in the editor | Program must be running with java  2)”File” option must be clicked  3) “Save” option should be selected | - | 1)Click on File tab on menu bar.  2)Click on “Save” option | A dialog box must appear which gives user the options to save file from computer | A dialog box must appear which gives user the options to save file from computer | PASS |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 2\_5 | Test if user is able to Print operation in the editor | Program must be running with java  2)”File” option must be clicked  3) “Print” option must be selected | - | 1)Click on File tab on menu bar.  2)Click on “Print” option | A dialog box must appear which gives user the option to Print  file from computer | A dialog box appear  which gives user the option to Print  file from computer | PASS |

**MODULE 3 – (Print Options)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case\_id** | **Test Case** | **Precondition** | **Test Data** | **Steps** | **Expected Output** | **Actual Output** | **Result** |
| 3\_1 | Test if user is able to open and view print options | Program must be running with java  2)”File” option must be clicked  3) “Print” option must be selected | - | 1)Click on File tab on menu bar.  2)Click on “Print” option | A dialog box must appear which gives user the option to Print  file from computer | A dialog box appear which gives user the option to Print  file from computer | PASS |
| 3\_2 | Test if user is able to do general changes in printing | Program must be running with java  2)”File” option must be clicked  3) “Print” option must be selected  4)”General” Tab must be visible | - | 1)Click on File tab on menu bar.  2)Click on “Print” option  3)Select “General”  Tab. | A dialog box must appear which gives user the options  To change general settings of printing | A dialog box appear which gives user the options  To change general settings of printing | PASS |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3\_3 | Test if user is able to do page setup in printing | Program must be running with java  2)”File” option must be clicked  3) “Print” option must be selected  4)”Page Setup” Tab must be visible | - | 1)Click on File tab on menu bar.  2)Click on “Print” option  3)Select “Page Setup”  Tab. | A dialog box must appear which gives user the options  to change page setup of printing | A dialog box appear which gives user the options  to change page setup of printing | PASS |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3\_4 | Test if user is able to do appearance settings in printing | Program must be running with java  2)”File” option must be clicked  3) “Print” option must be selected  4)”Appearance” Tab must be visible | - | 1)Click on File tab on menu bar.  2)Click on “Print” option  3)Select “Appearance”  Tab. | A dialog box must appear which gives user the options  to change appearance of page while printing | A dialog box appear which gives user the options  to change appearance of page while printing | PASS |

**MODULE 4 – (Edit Options)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case\_id** | **Test Case** | **Precondition** | **Test Data** | **Steps** | **Expected Output** | **Actual Output** | **Result** |
| 4\_1 | Test if user is able to open and view edit options | Program must be running with java | - | 1)Click on Edit tab on menu bar | Various Different Options must be available | Various Different Options are  available | PASS |
| 4\_2 | Test if user is able to use cut function in menu | Program must be running with java  2) “Edit” option must be clicked | “hello” | 1)Click on Edit tab on menu bar  2)Click on “Cut” option | The inputted string should be cut. | The inputted string is cut. | PASS |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 4\_3 | Test if user is able to use copy function in menu | Program must be running with java  2) “Copy” option must be clicked | “copy” | 1)Click on Edit tab on menu bar  2)Click on “Copy” option | The inputted string should be copied. | The inputted string is copied. | PASS |
| 4\_4 | Test if user is able to use copy function in menu | Program must be running with java  2) “Copy” option must be clicked | “copy” | 1)Click on Edit tab on menu bar  2)Click on “Copy” option | The inputted string should be copied. | The inputted string is copied. | PASS |

**Module 5 – (Exit Option)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case\_id** | **Test Case** | **Precondition** | **Test Data** | **Steps** | **Expected Output** | **Actual Output** | **Result** |
| 5\_1 | Test if user is able to Exit Program | Program must be running with java  2) “Exit” option must be clicked | - | 1)Click on Exit tab on menu bar | Program must be closed | Program must be closed | PASS |

**Chapter-6**

**CONCLUSION**

Although software testing can determine the correctness of software under the assumption of some specific hypotheses (see the hierarchy of testing difficulty below), testing cannot identify all the failures within the software. Instead, it furnishes a criticism or comparison that compares the state and behavior of the product against test oracles — principles or mechanisms by which someone might recognize a problem. These oracles may include (but are not limited to) specifications, contracts, comparable products, past versions of the same product, inferences about intended or expected purpose, user or customer expectations, relevant standards, applicable laws, or other criteria.

A primary purpose of testing is to detect software failures so that defects may be discovered and corrected. Testing cannot establish that a product functions properly under all conditions, but only that it does not function properly under specific conditions. The scope of software testing may include the examination of code as well as the execution of that code in various environments and conditions as well as examining the aspects of code: does it do what it is supposed to do and do what it needs to do. In the current culture of software development, a testing organization may be separate from the development team. There are various roles for testing team members. Information derived from software testing may be used to correct the process by which software is developed.

Every software product has a target audience. For example, the audience for video game software is completely different from banking software. Therefore, when an organization develops or otherwise invests in a software product, it can assess whether the software product will be acceptable to its end users, its target audience, its purchasers, and other stakeholders. Software testing assists in making this assessment.

**Chapter-7**

**References**

1. https://en.wikipedia.org/wiki/Software\_testing

2. https://salsadigital.com.au/insights/a-brief-history-of-software-testing

3. https://en.wikipedia.org/wiki/Java\_(programming\_language)

4. https://www.google.com/search?q=Glenford+J.+Myers

5. https://www.google.com/search?q=software+testing+