VERSION

1

JAVA OBFUSCATOR HELP MANUAL

HELPMANUAL

JAVA OBFUSCATOR HELP MANUAL

TABLE OF CONTENTS

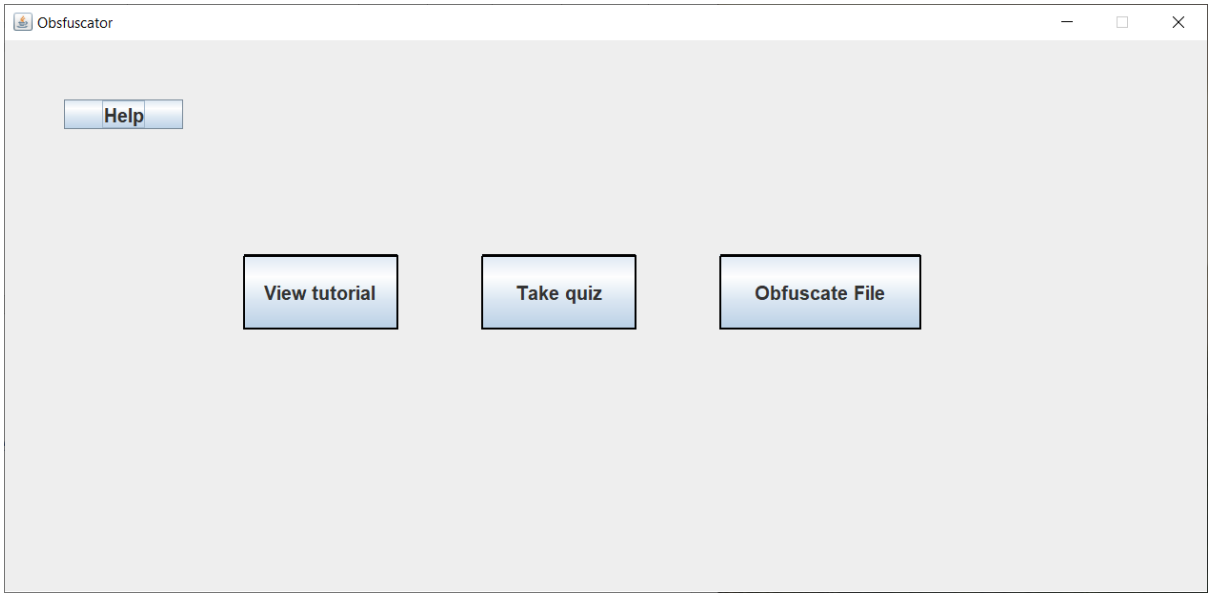
Table of Contents

1. User Interface Navigation

1.1 Menu

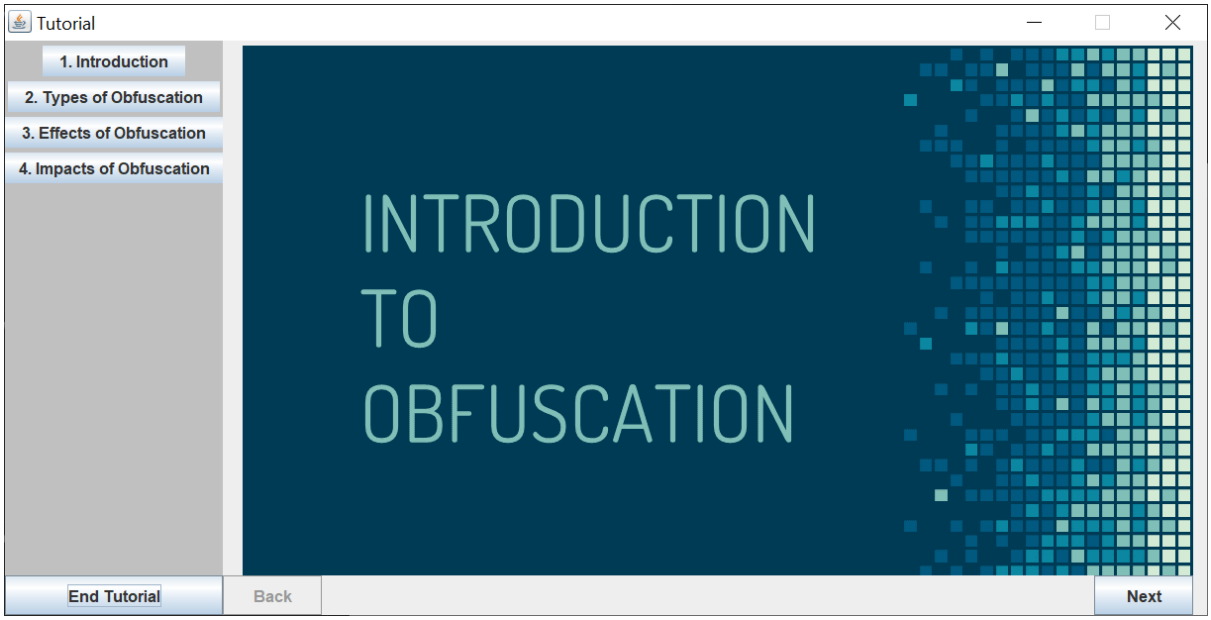
At this stage, the user has successfully launched the obfuscator application and clicked on the technical menu. There are 3 options to proceed with.

1) View tutorial – This would launch a tutorial within the app itself. The tutorial contains information regarding all you need to know about obfuscation. This would enhance the user with the knowledge to prepare to take the quiz (following option)
2) Take Quiz – A series of multiple-choice questions that records your high score upon completion! Questions are regarding obfuscation and java. You will be able to ace it once you've gone through the tutorial!
3) Obfuscate File – This is where you put your knowledge into practical use. You may input a java file into the application and you it will output an obfuscated file based on the methods you desire!
4) Help Function – The "Help" button when clicked, opens a help manual in pdf format.

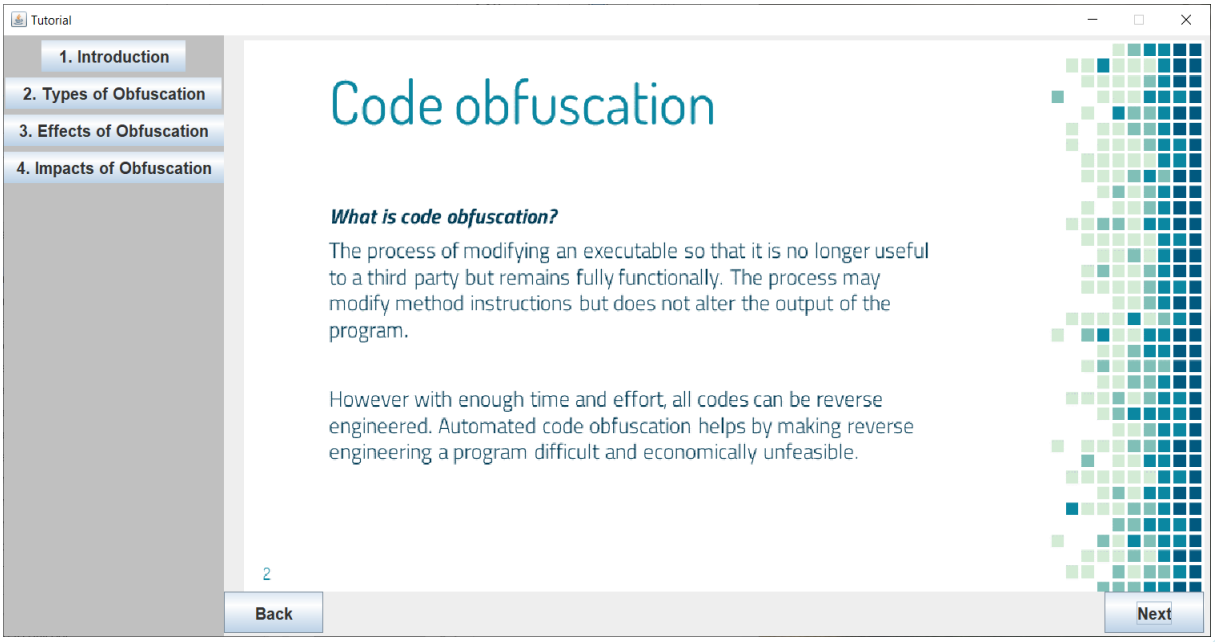


1.2 View Tutorial

In this tutorial, the chapters are nicely organized on the left side bar. You may skip to the latest chapter you left off with just by clicking on the title bar. Eg. “Effects of obfuscation” Once you are done with reading and understand the tutorial page content, you may click on the next button to proceed to the next page.



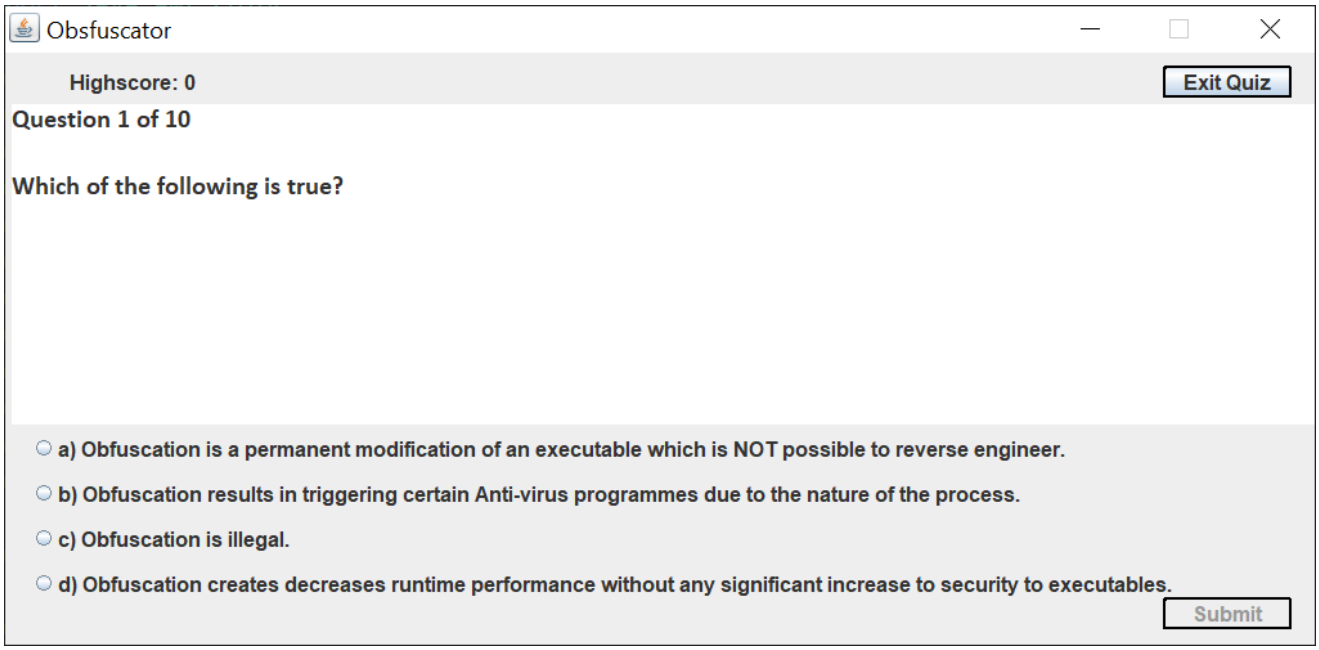
Just like any other tutorial, our slides are compressed into simple and easy to understand content. This provides the user the comprehension of ‘the last tutorial youll ever need’.
By clicking on the next button, a new page will be shown and youll be one page lesser to the ending.
The back button enables the user to read back on the previous pages should any doubt arises.



1.3 Take Quiz

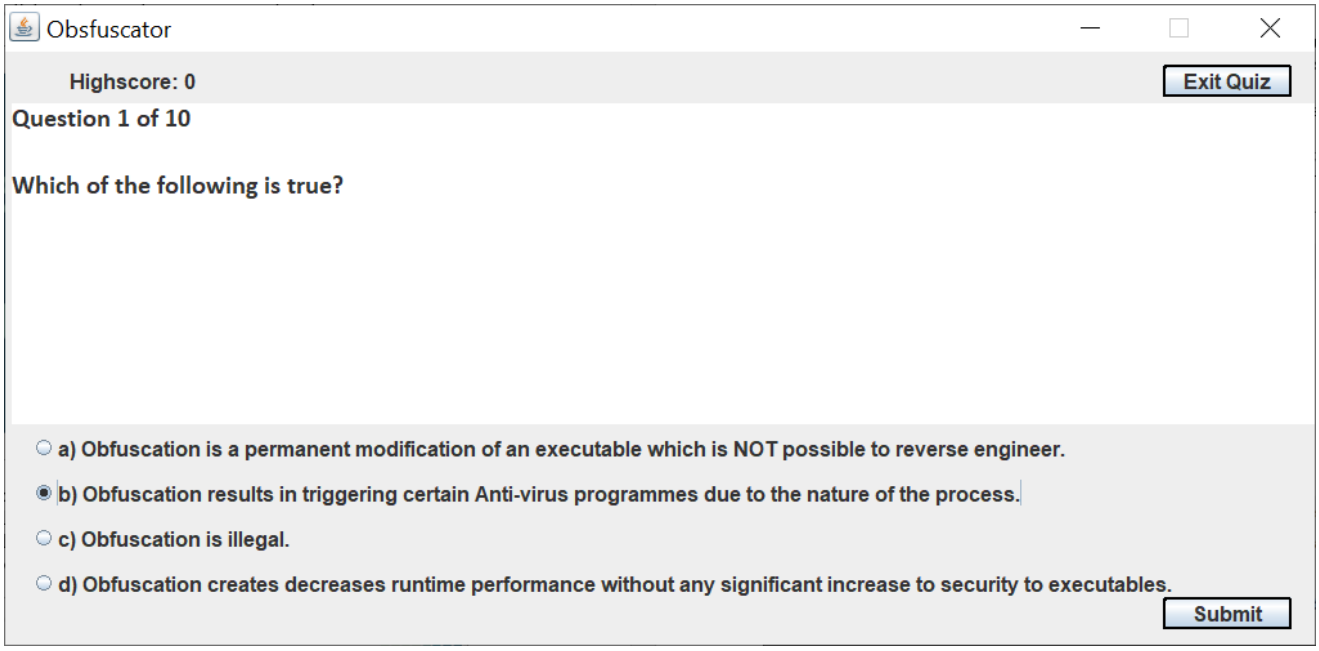
1.3.1 Quiz Display

By clicking on the 'Take Quiz' button, you will end up opening the quiz. In this section, you will be able to pit yourself with the knowledge of obfuscation. You may challenge yourself to get the highest score possible! Any doubts or questions can be answered by referring to our tutorial!
Multiple choice questions. There are 4 options available for each question. You may pick the best answer to your knowledge. Upon selection, you may submit your answer.
Should a break or event pop up, and you are unable to continue with the quiz, you may click on the exit quiz to terminate it. No high score will be recorded when you exit the quiz prematurely.



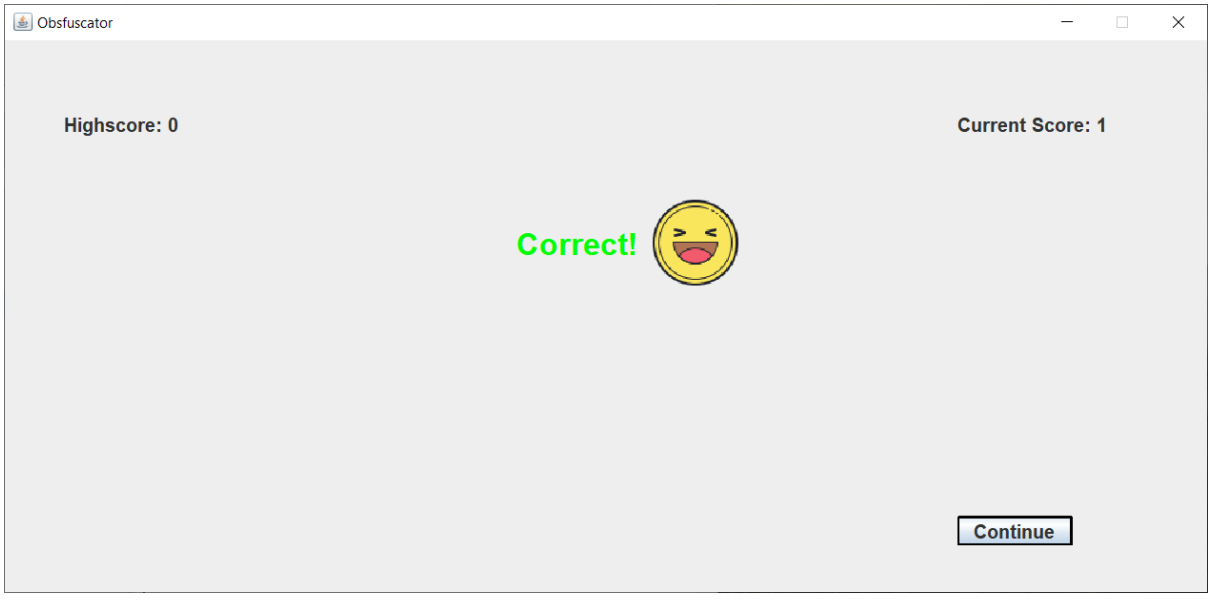
1.3.2 Submit Quiz Answer

The submit button will only be enabled when an answer is selected. Otherwise it will be greyed out. Please select the most appropriate answer to your knowledge



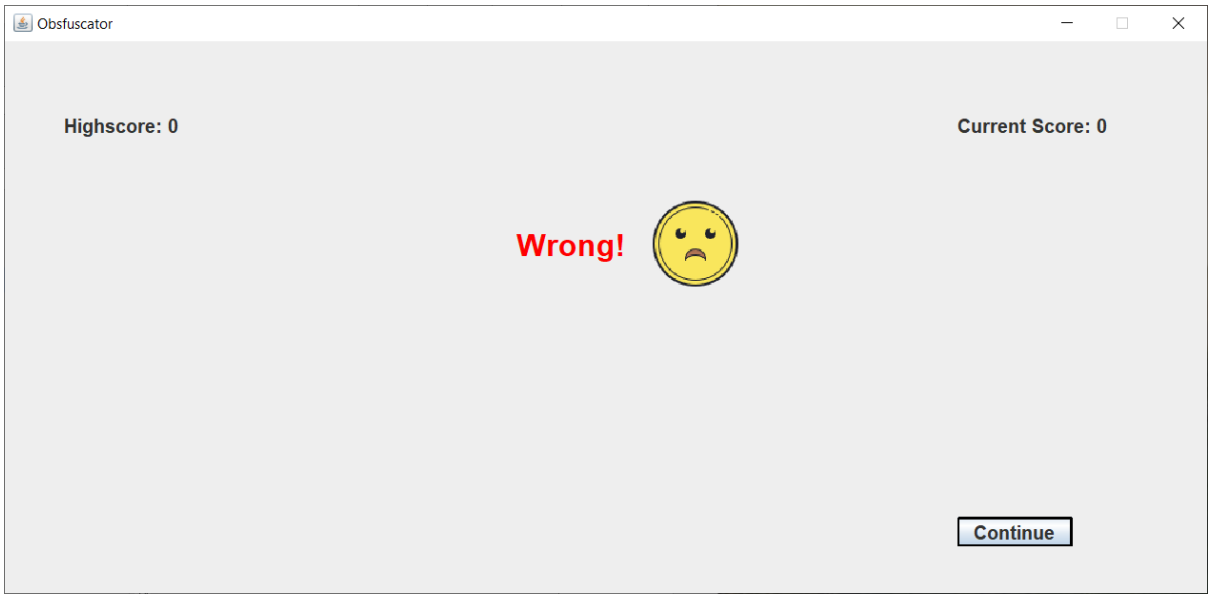
1.3.3 Correct Answer

Upon selecting the correct answer, the current score will be updated with +1. You will be greeted with a smiley face. High score is only updated at the end of the quiz. You may click on continue button to proceed to the next question.



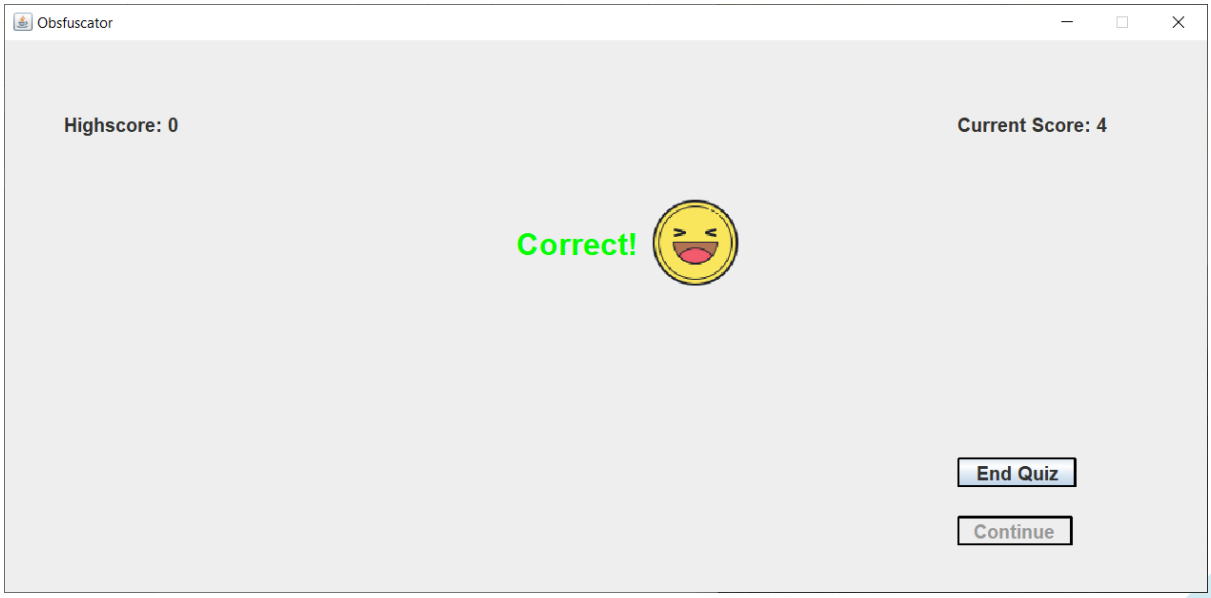
1.3.4 Wrong Answer

Selecting the wrong answer will display a sad smiley face. This is to encourage you to work harder and ace this test scoring the highest! You may click on continue button to proceed to the next question. All the best for the next question!



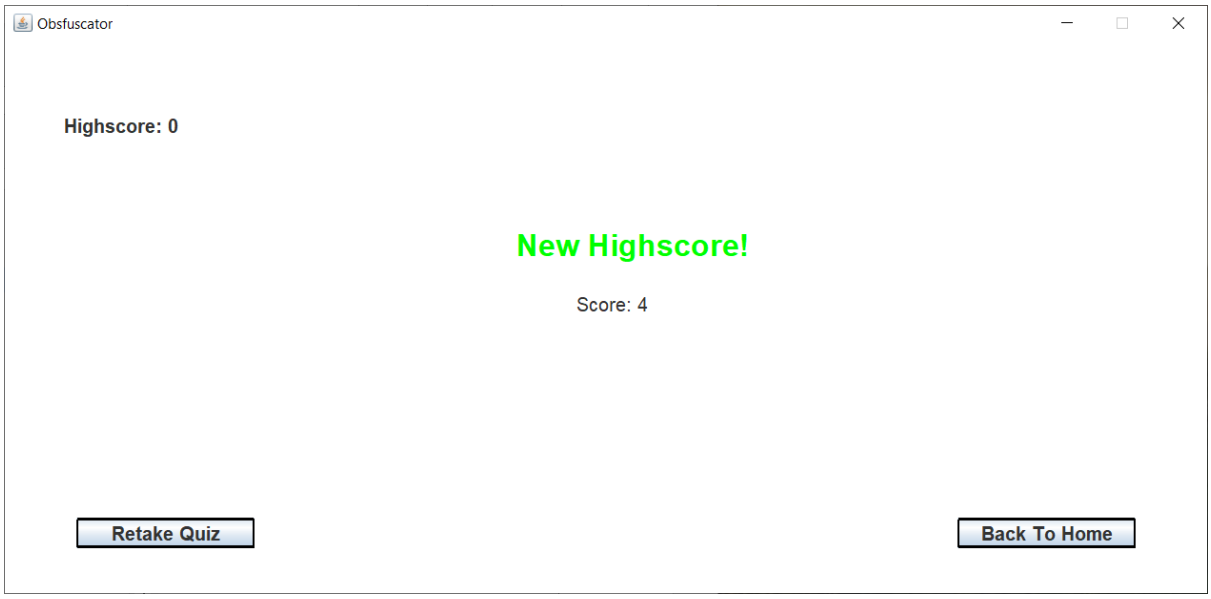
1.3.5 Last Page of Quiz

The continue button is greyed out as there is no further questions to be quizzed upon.
After finishing the last question of the quiz, the End quiz button will be shown. The current score will be updated as your new high score only if it is higher than the current highest score.



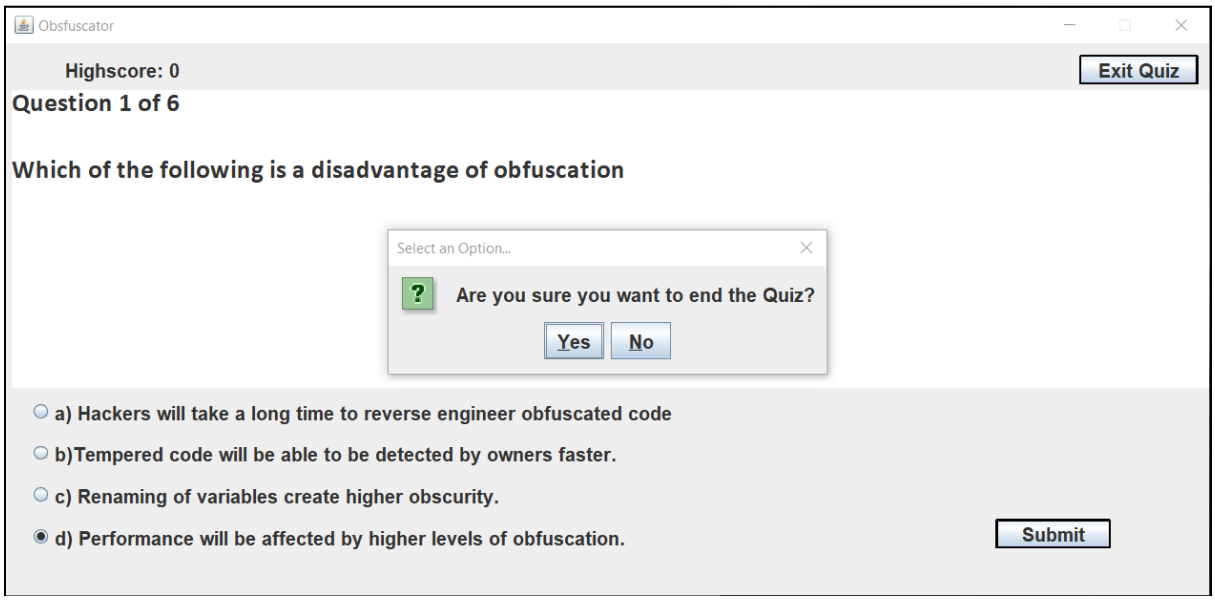
1.3.6 Quiz Highscore

You may be able to retake the quiz to challenge yourself to beat your previous high score or the current high score. Your total score will be shown on the screen.
User may return back to the home page to use other function within the app.



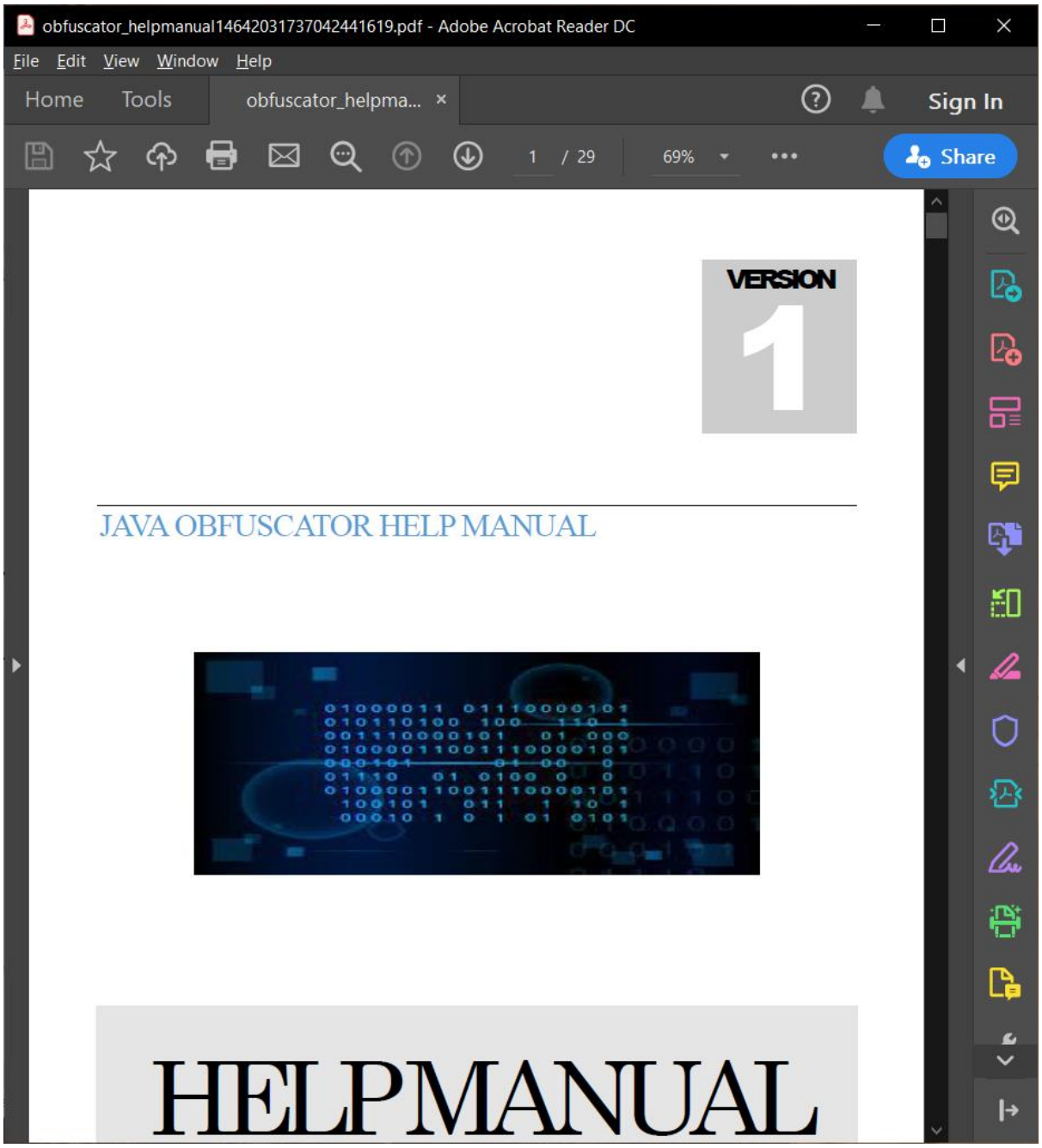
1.3.7 Exit Quiz

User may exit the quiz anytime. Upon clicking the quiz there would be another verification if you are sure that this is your decision. This is to prevent accidental clicking of the button and ending the quiz.



1.4 Help Function

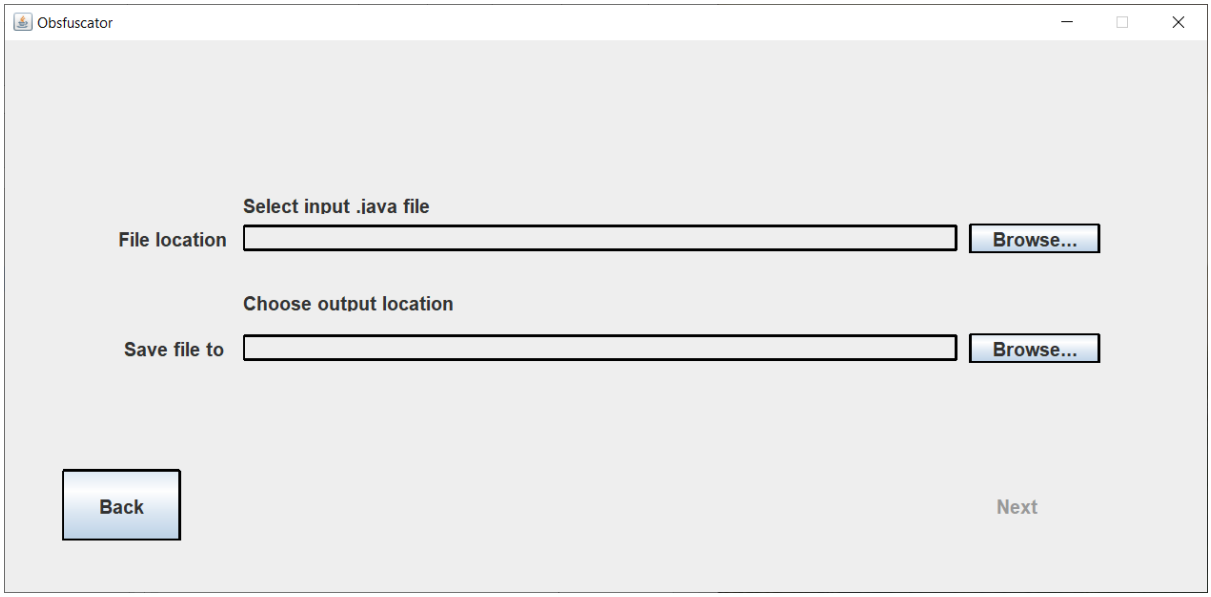
The “Help” button located in the main menu opens a help manual in .PDF format. It explains the instructions on how to use the obfuscator and obfuscation techniques available in the obfuscator.



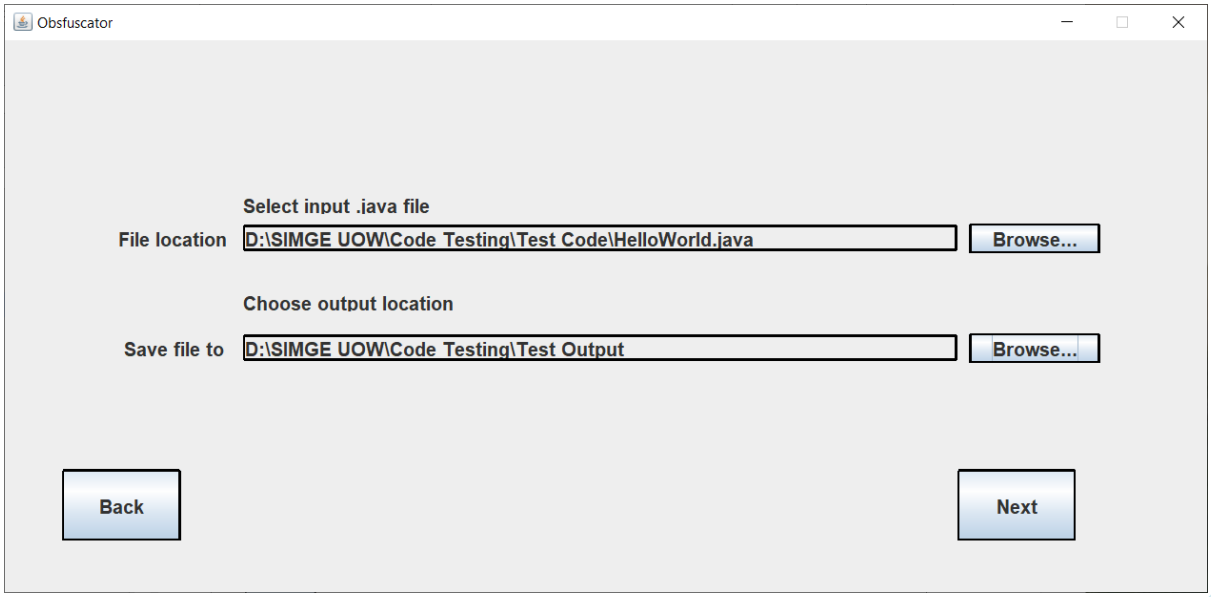
1.5 Obfuscate File

1.5.1 Select Input File and Output Directory

Please input the file with the extension of .java only. You may click on the browse button to search up the directory of where your desired files are.
Subsequently you may choose the folder you wish to have the output files saved at.
If this function (obfuscate a file) was clicked accidently, you may click the back button to show the previous page.

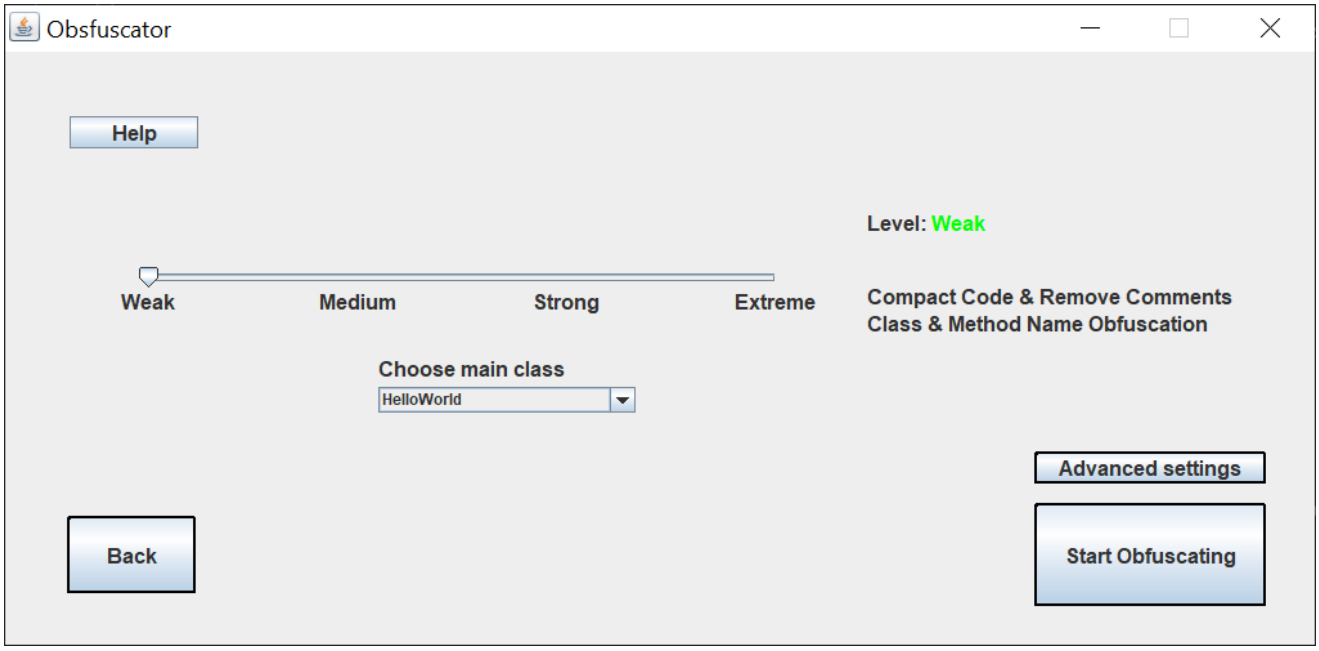


After filling up the necessary fields, you will be able to advance to the next step by clicking the next button.



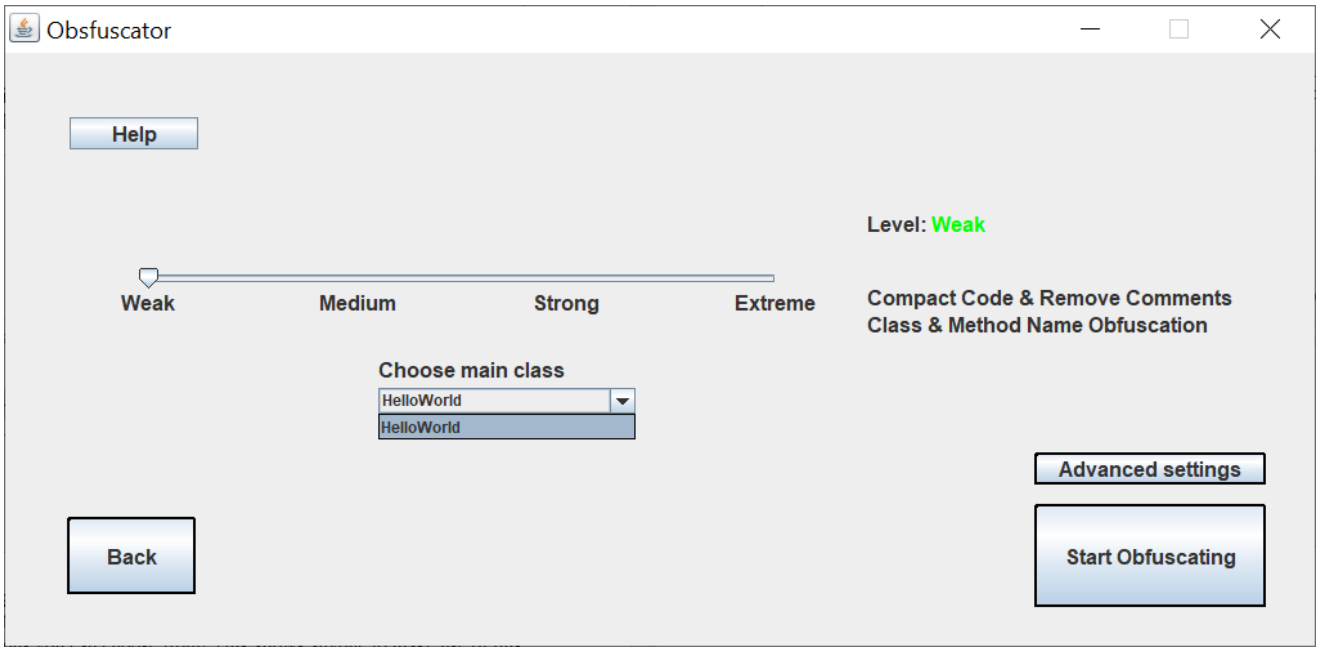
1.5.2 Basic Obfuscation Settings

This is the basic setting panel. This panel is most suitable for users that wish to do a quick / fast obfuscation. The obfuscations methods used are defined in a preset (Weak, Medium, Strong, Extreme)



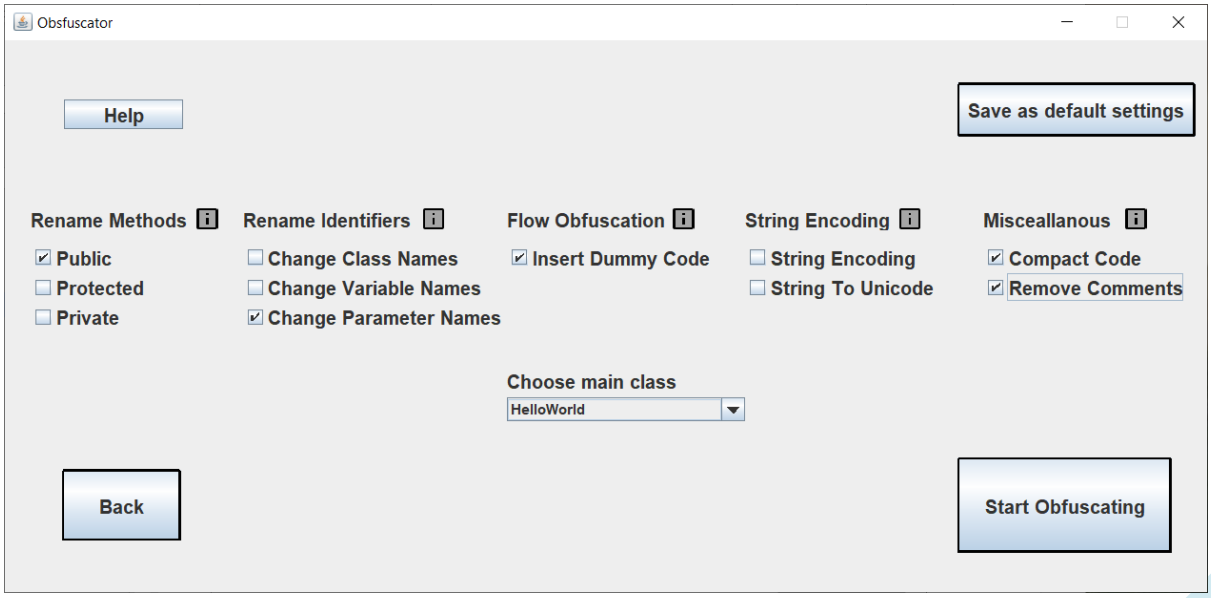
1.5.3 Choose Main Class

Choose a main class from the settings panel, this main class will be the name of your output file. Note that if you change the class name, the output file will be obfuscated version of the main class. Example HelloWorld.java will be changed to l1l1lIlI1lI.java.



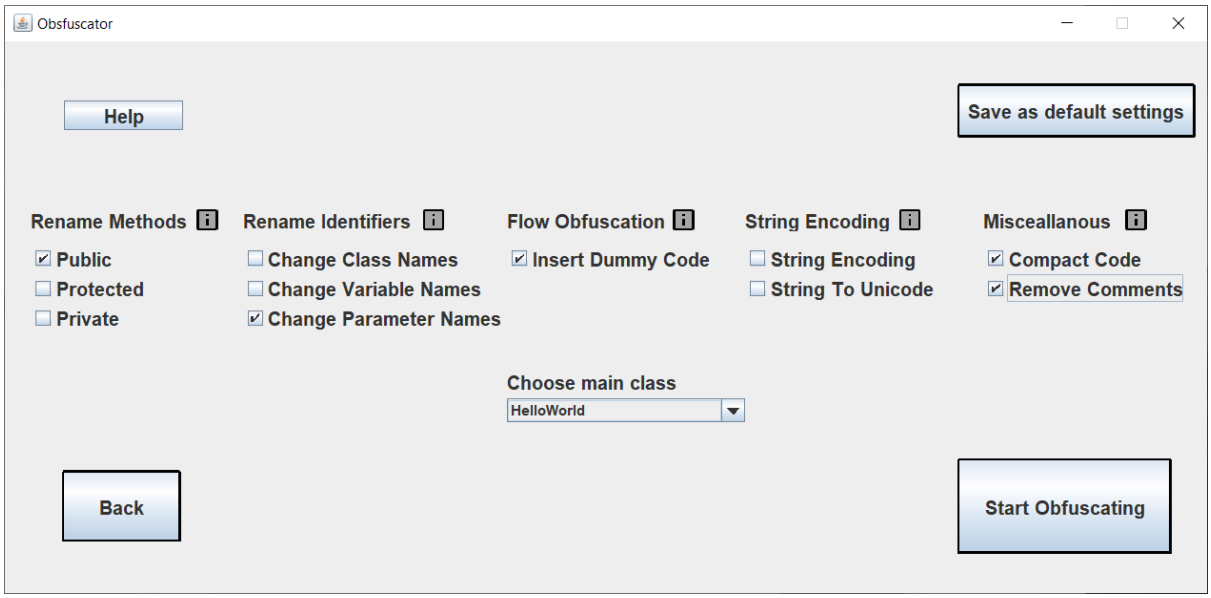
1.5.4 Advanced Obfuscation Settings

The advance obfuscation setting page will enable the user to customized the obfuscation methods used for the file. There can be up to 61 different combinations you can choose from! This allows anyone to make use of this function to actually learn more in depth about each obfuscation techniques and how they complement each other

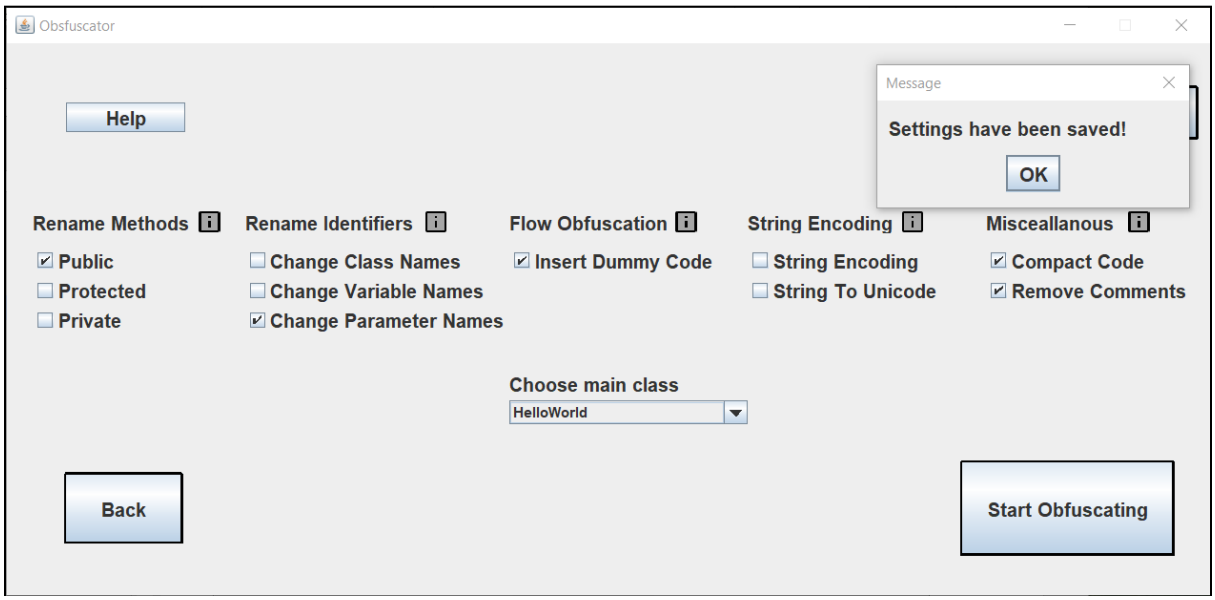


1.5.5 Save Default Advanced Settings

The save as default setting helps speed up redundant time selecting the most commonly used techniques a user may choose. This way, each time the program is run, the preferred settings will be checked and ready to obfuscate.

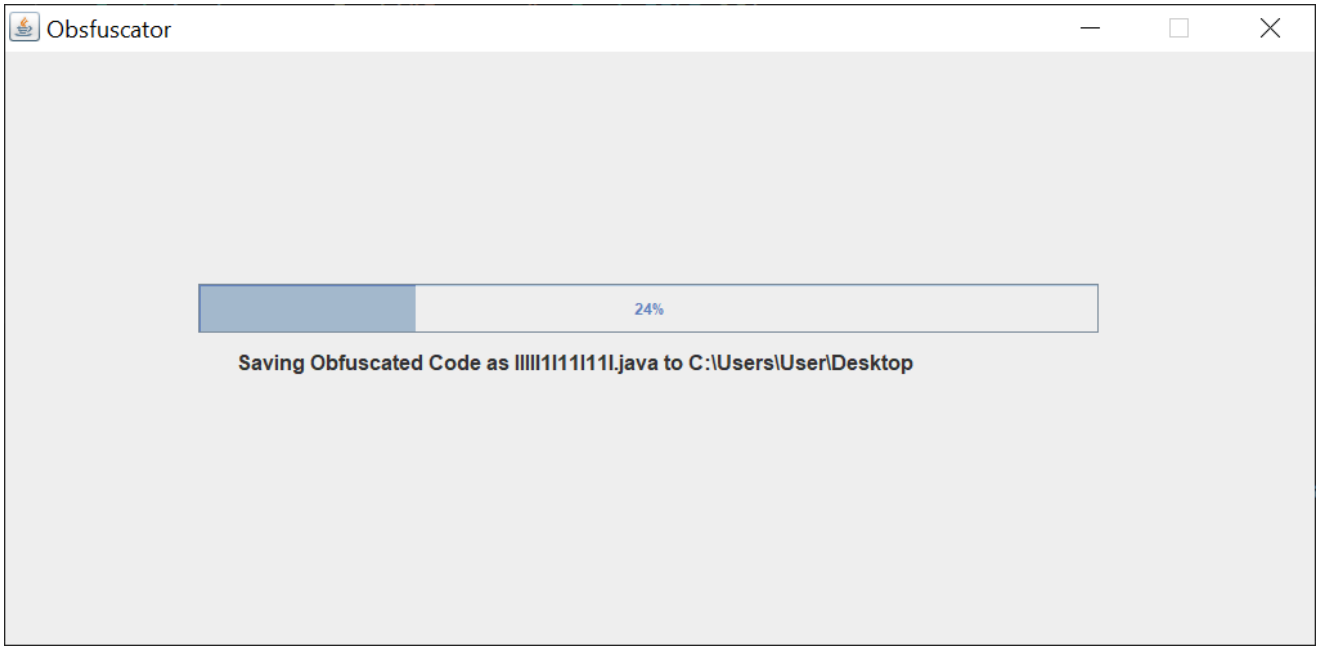


Once the setting is clicked, a message box would appear stating so. After which your setting will be saved upon next session. You may click ok to proceed.



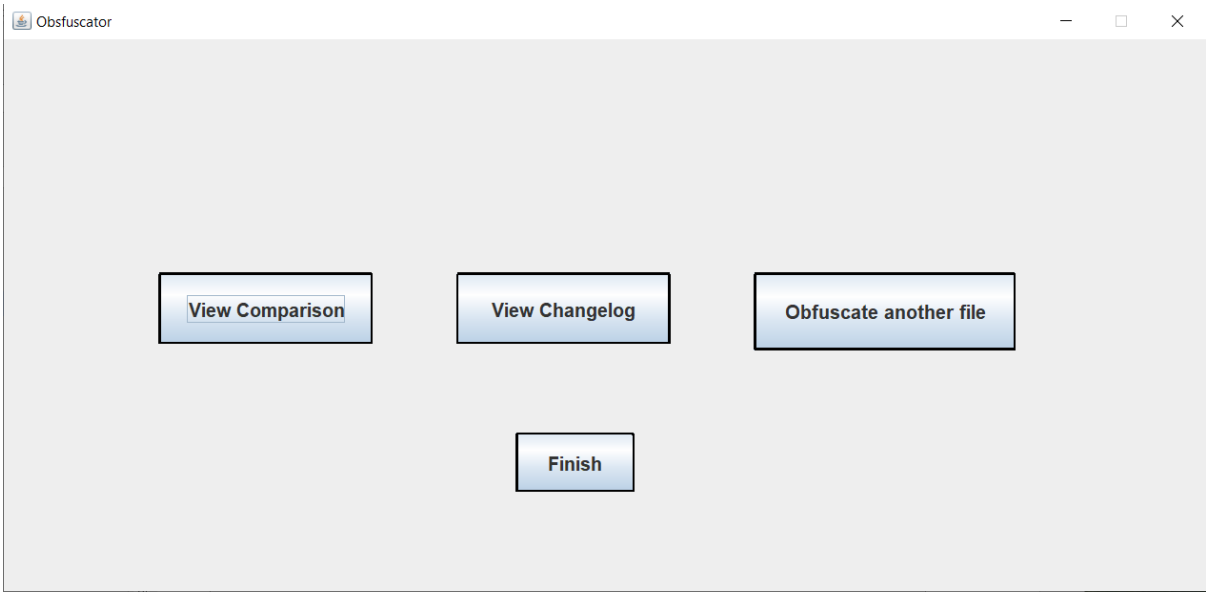
1.5.6 Obfuscate Input File

While the file is being obfuscated, the progress bar will be shown. The output location is also shown below the bar. The bar is an approximate duration the application takes.



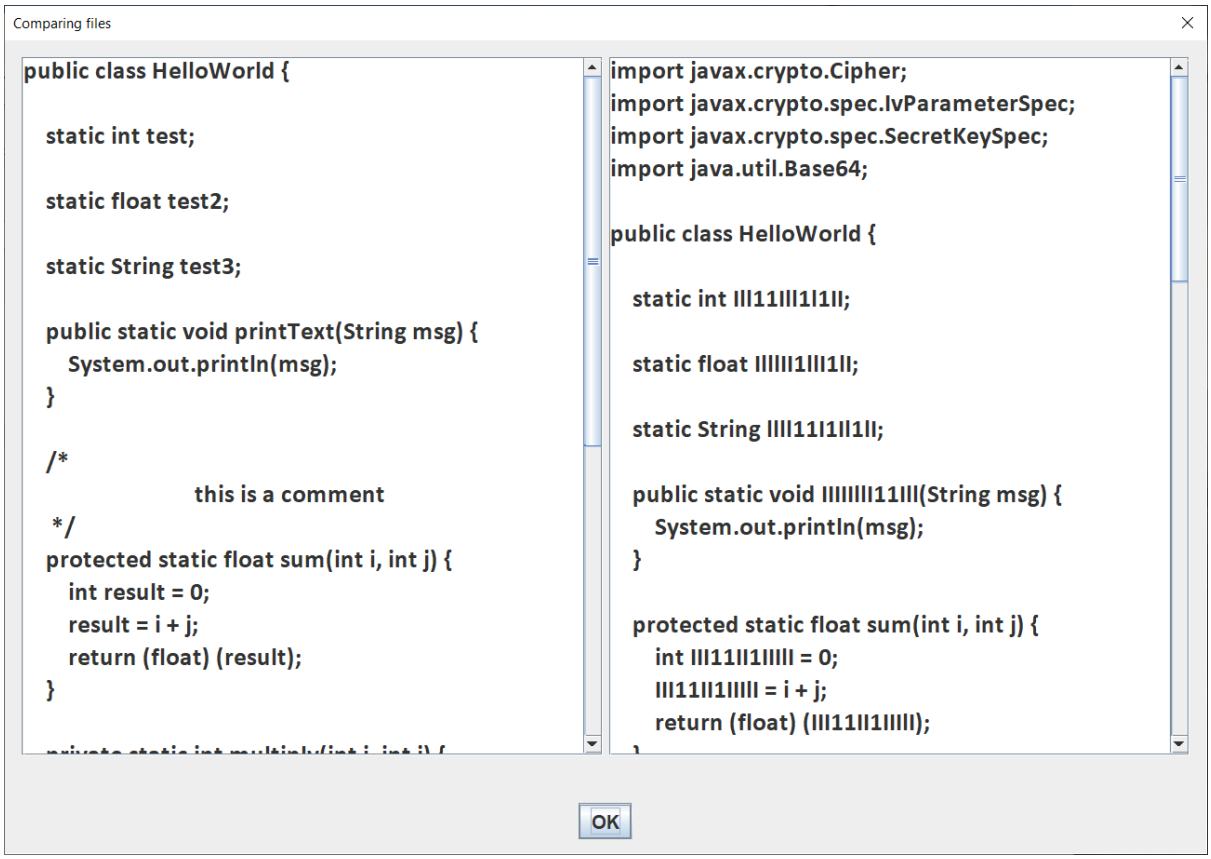
1.6 Post-Obfuscation Menu

Once the file finish obfuscating, the menu is shown as such. There are 4 options to proceed with.
View Comparison – It will show the file before and after obfuscation side by side to use for comparison and understanding the code better.
View changelog – This changelog explains to the user what has been done to the code in detail. Eg: Rename of variable to a different name and it is replaced how many times.
Obfuscate another file – This option is to provide swift and easy access to obfuscate another file without the need to restart the application.
Finish – This button exits the program and closes it for you. To launch the program again, you may run it once again.



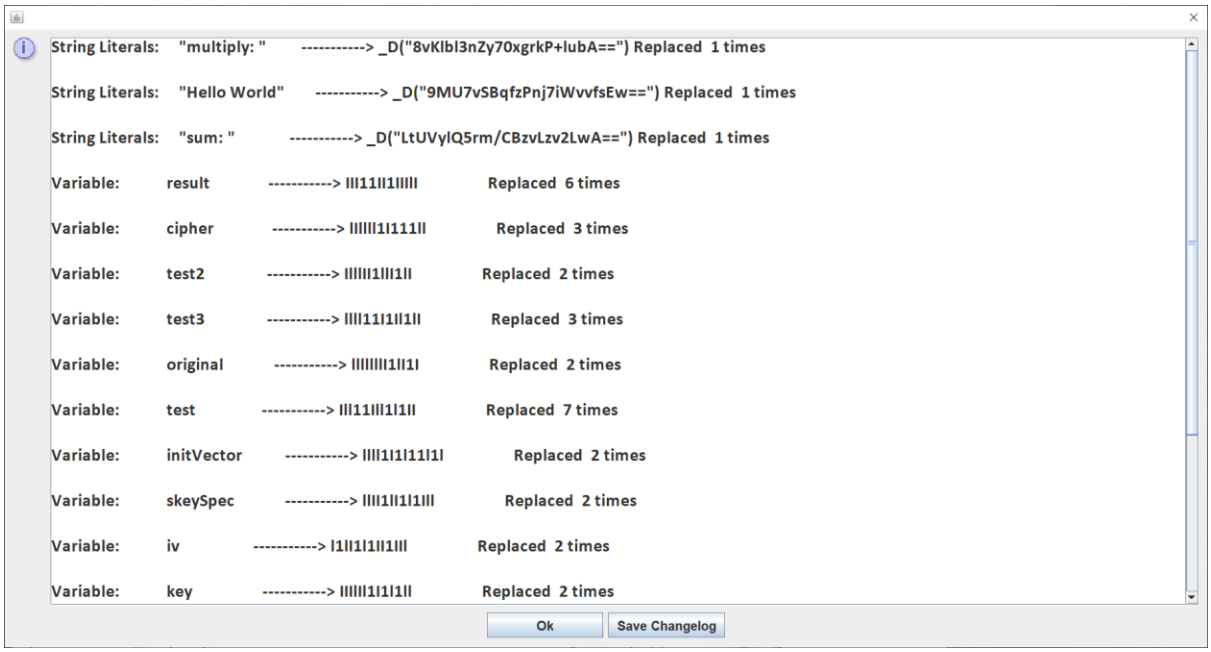
1.6.1 View Comparison

In this view, the before and after process is shown.
Left – Before obfuscation techniques are used
Right – After obfuscation techniques are used.



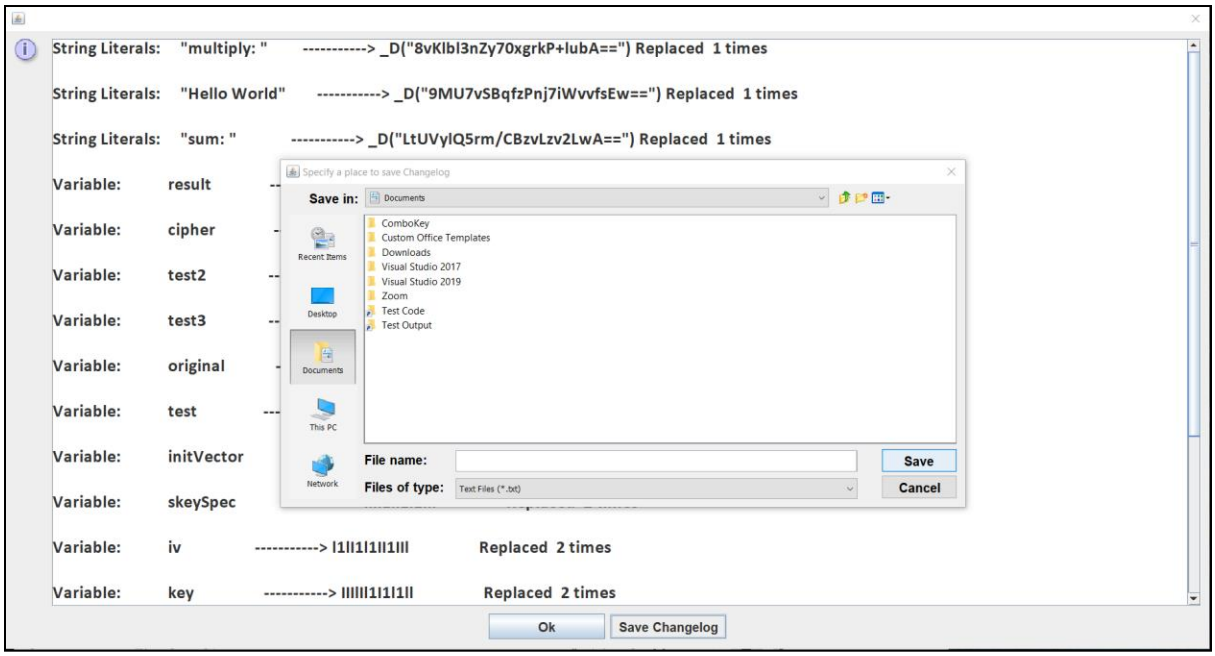
1.6.2 View Changelog

The change log shows details of what has been changed to the original file. Each change is documented as such, type of content, name of content and what it has been changed to. It also shows how many time it was changed throughout the file.



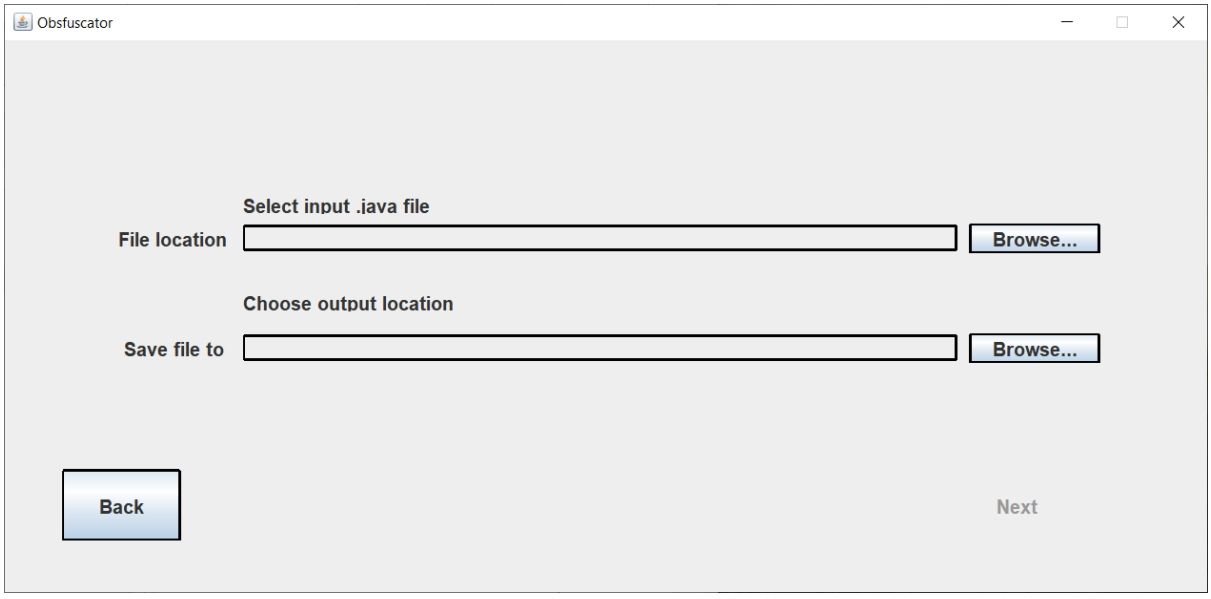
1.6.3 Save Changelog

To ensure that the user knows what has been changed to the file, you may also save the change log. This acts like a key to the obfuscated file. Please ensure that the file is safe from perpetrators if you are obfuscating this file for security reasons.
You may choose the output location of the changelog by navigating through the navigation.



1.6.4 Obfuscate Another File

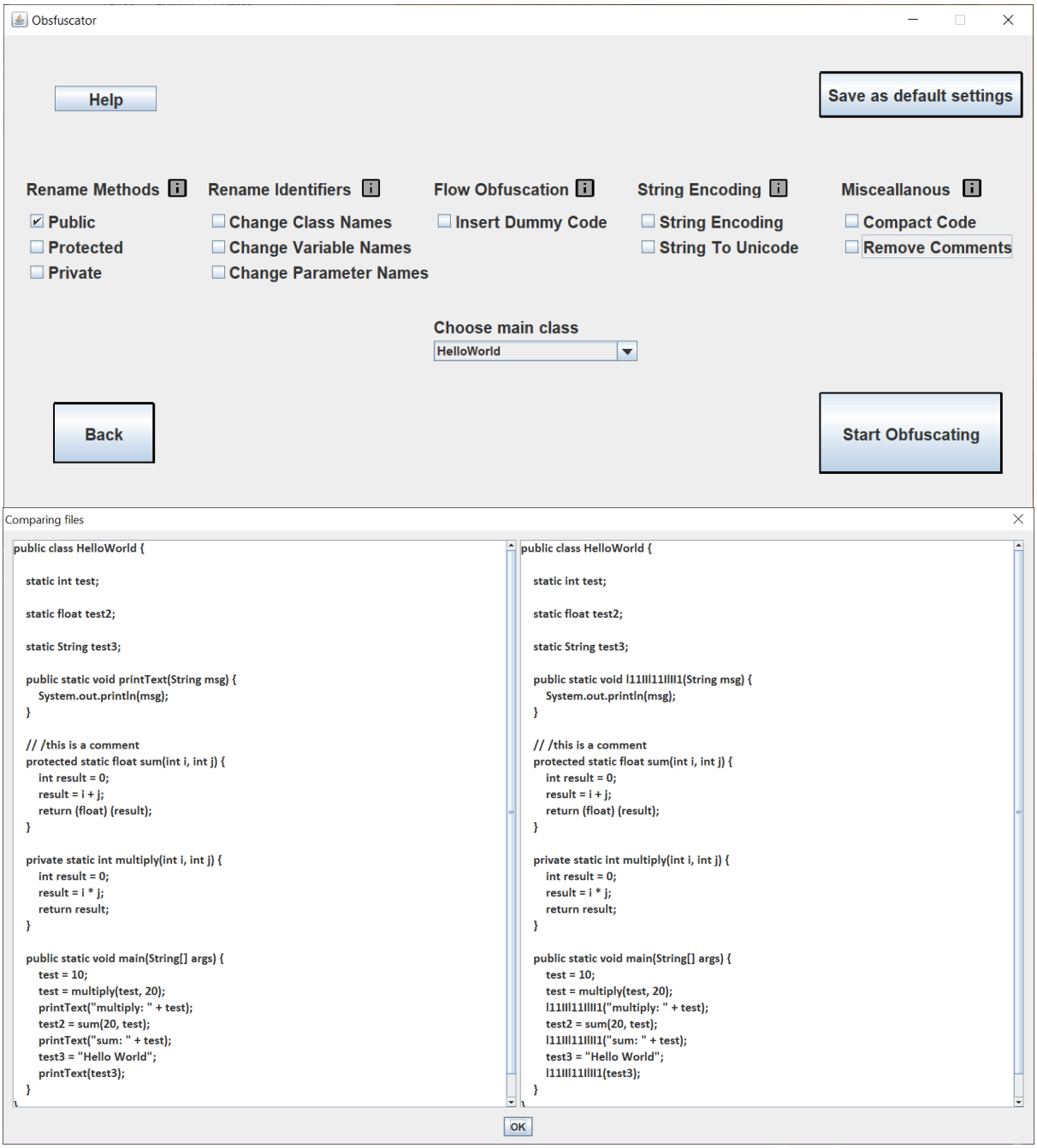
If obfuscate another file is clicked, the user will be brought back to the front page where a new file is needed for input to obfuscate. Follow the steps previously to start obfuscating again.



2. Obfuscation Features

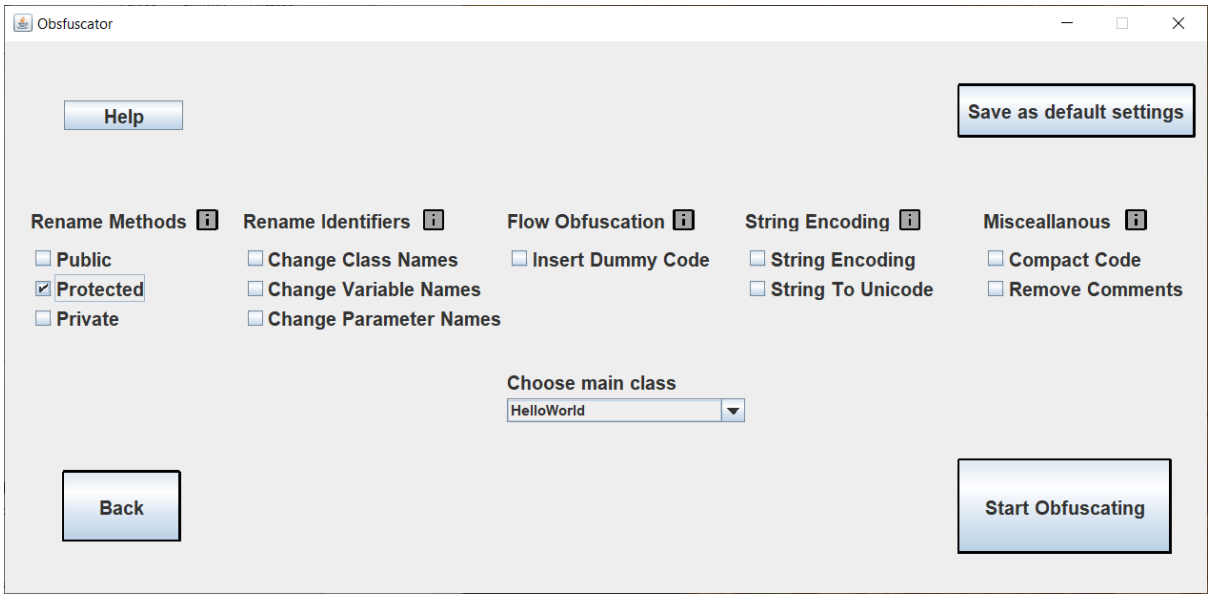
2.1 Rename Public Methods

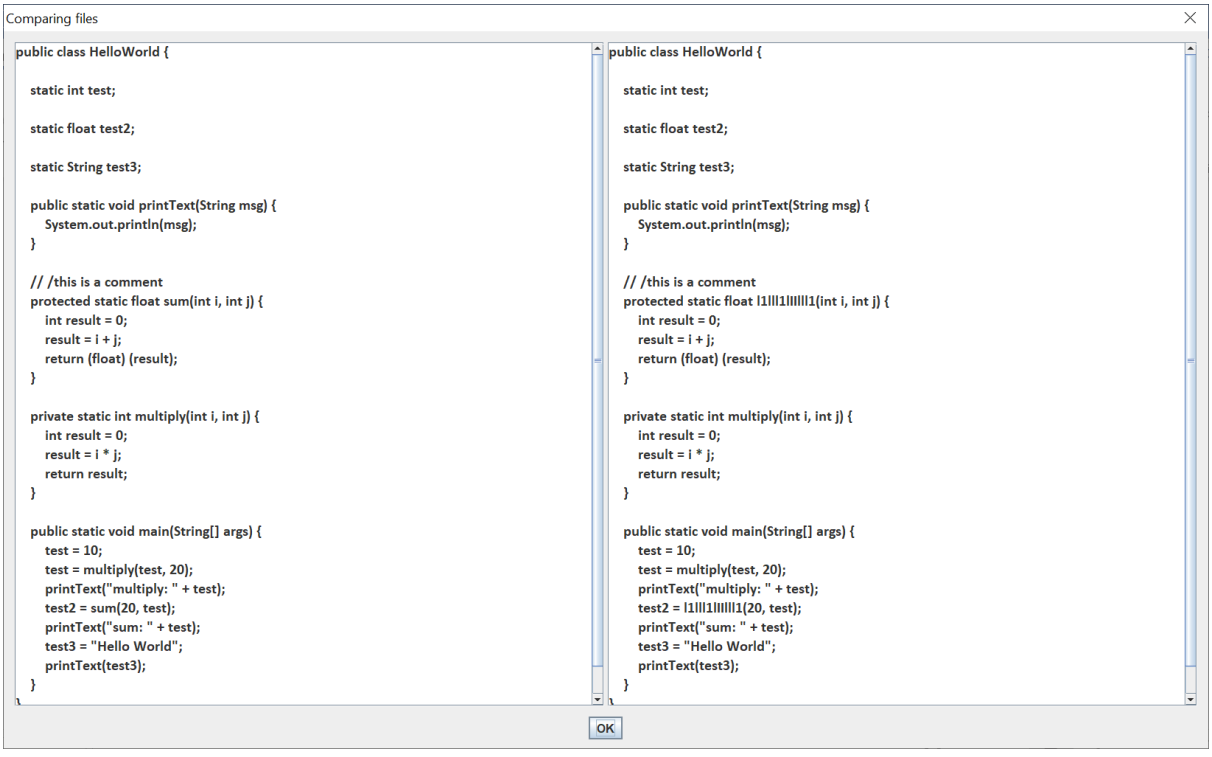
Using the technique of obfuscation, the public methods will have a change of name. Referring to this example below, you can see that the text ‘printText’ is being obfuscated. All ‘printText’ is being changed to undistinguishable texts.



2.2 Rename Protected Methods

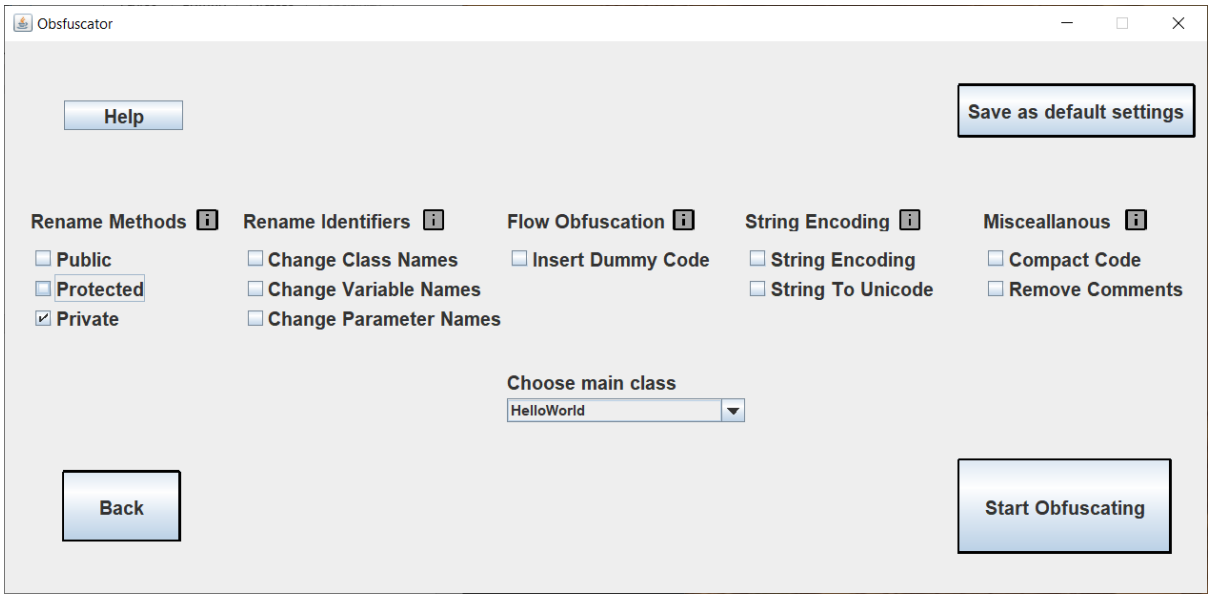
Using the technique of obfuscation, the protected methods will have a change of name. Referring to this example below, you can see that the text ‘sum’ is being obfuscated. All ‘sum’ is being changed to undistinguishable texts.

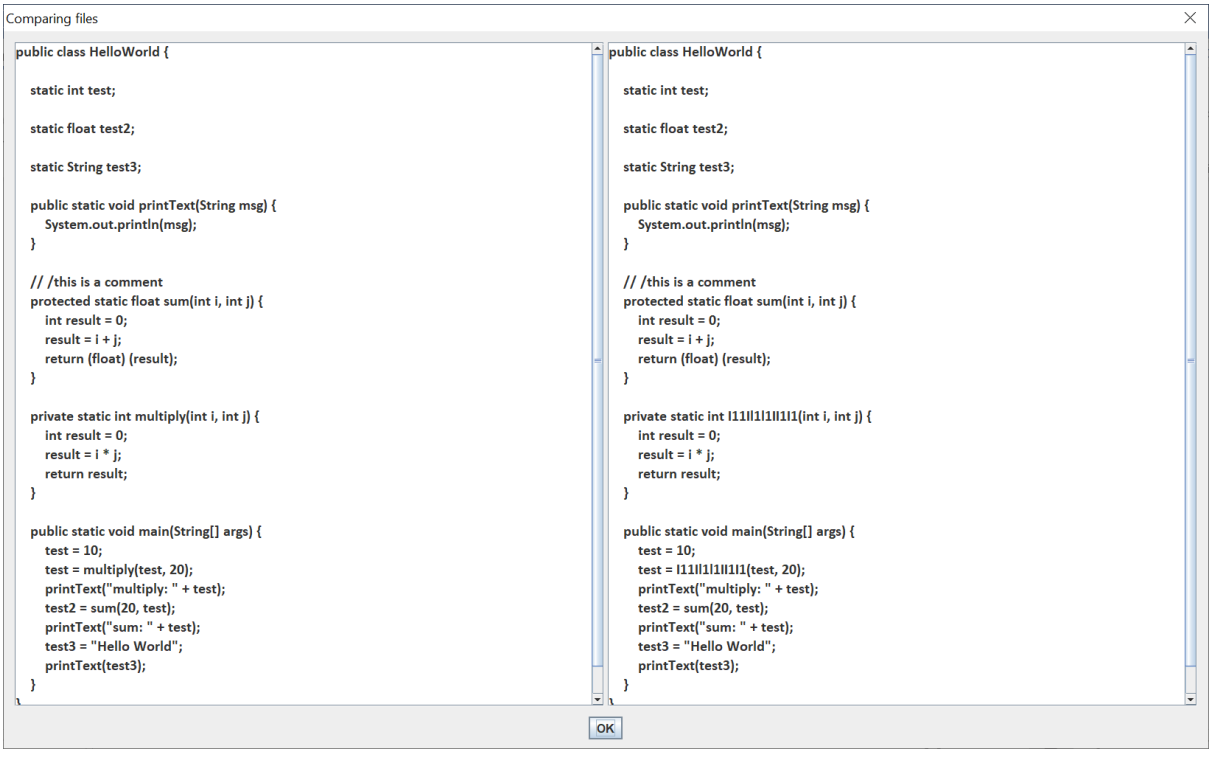




2.3 Rename Private Methods

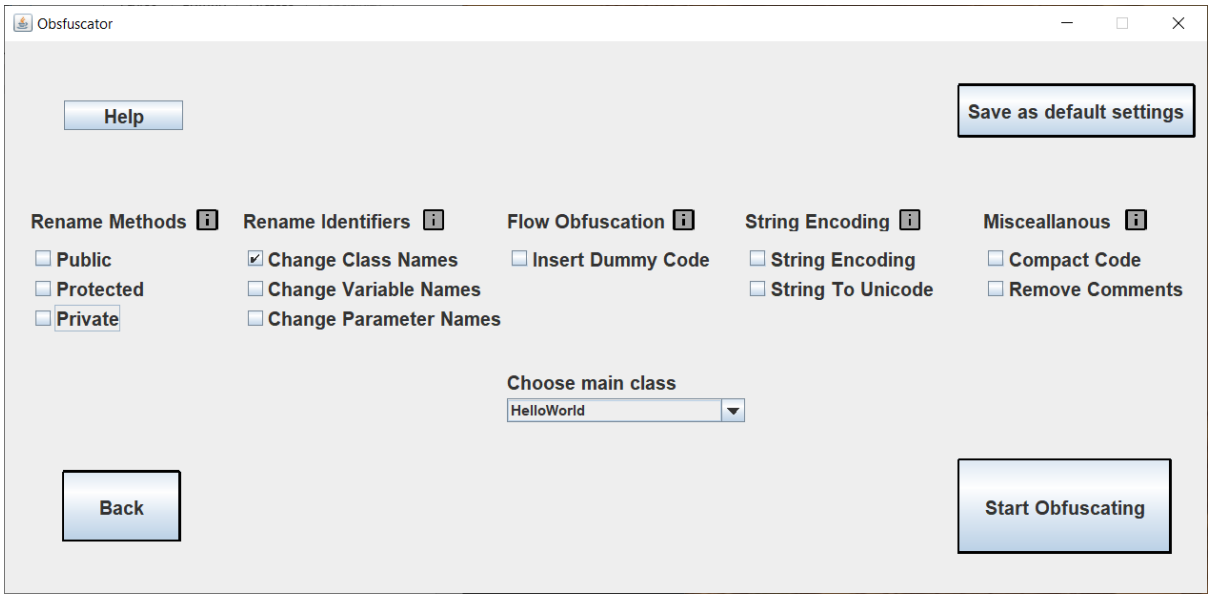
Using the technique of obfuscation, the private methods will have a change of name. Referring to this example below, you can see that the text ‘multiply is being obfuscated. All ‘multiply’ is being changed to undistinguishable texts.

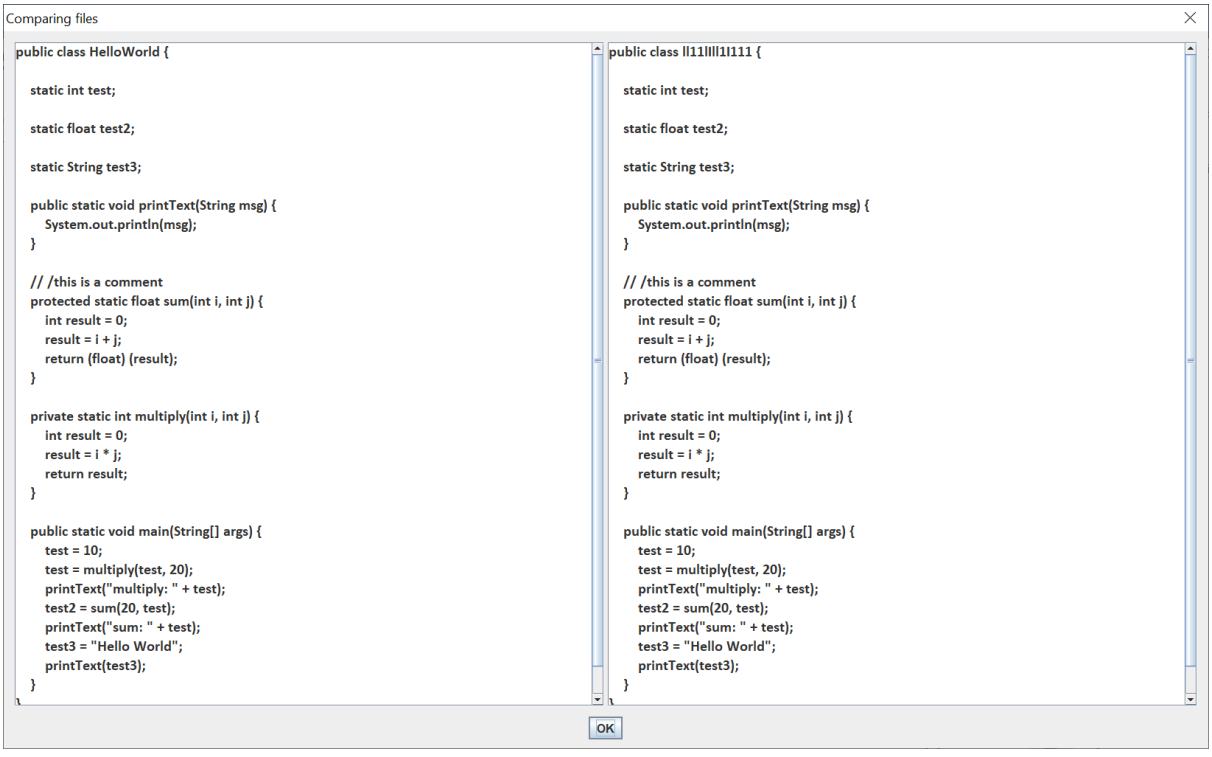




2.4 Change Class Name

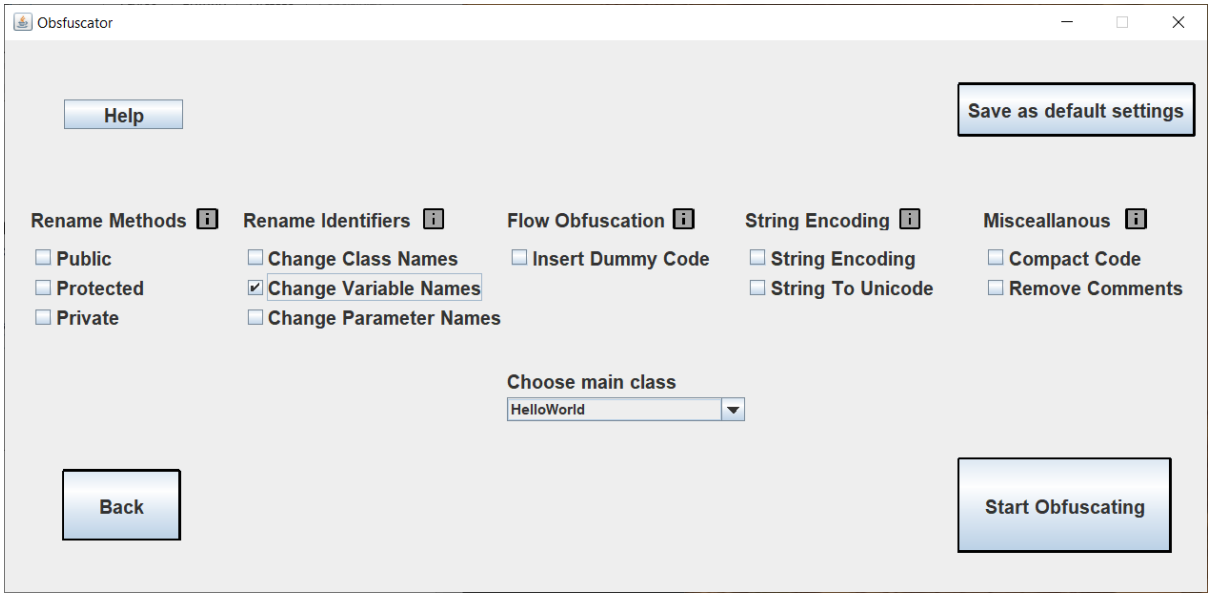
Using the technique of obfuscation, the class identifiers will have a change of name. Referring to this example below, you can see that the text ‘HelloWorld’ is being obfuscated. All ‘HellowWorld’ is being changed to undistinguishable texts.

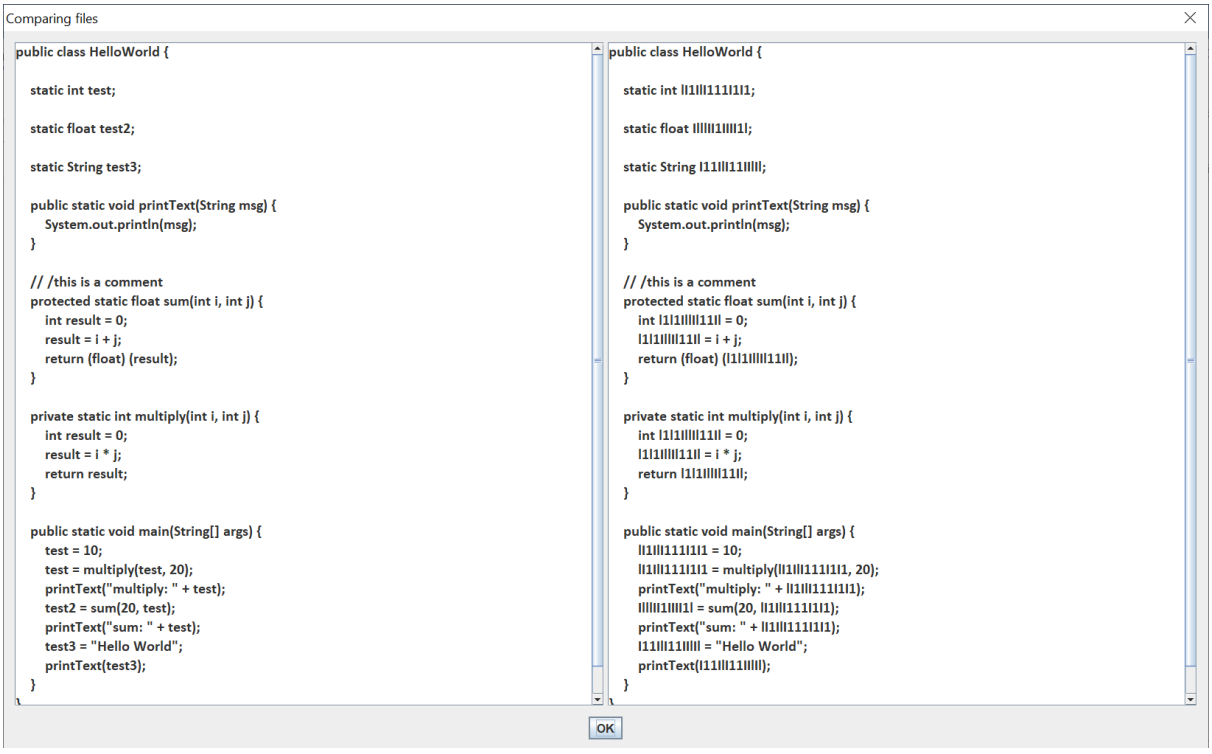




2.5 Change Variable Names

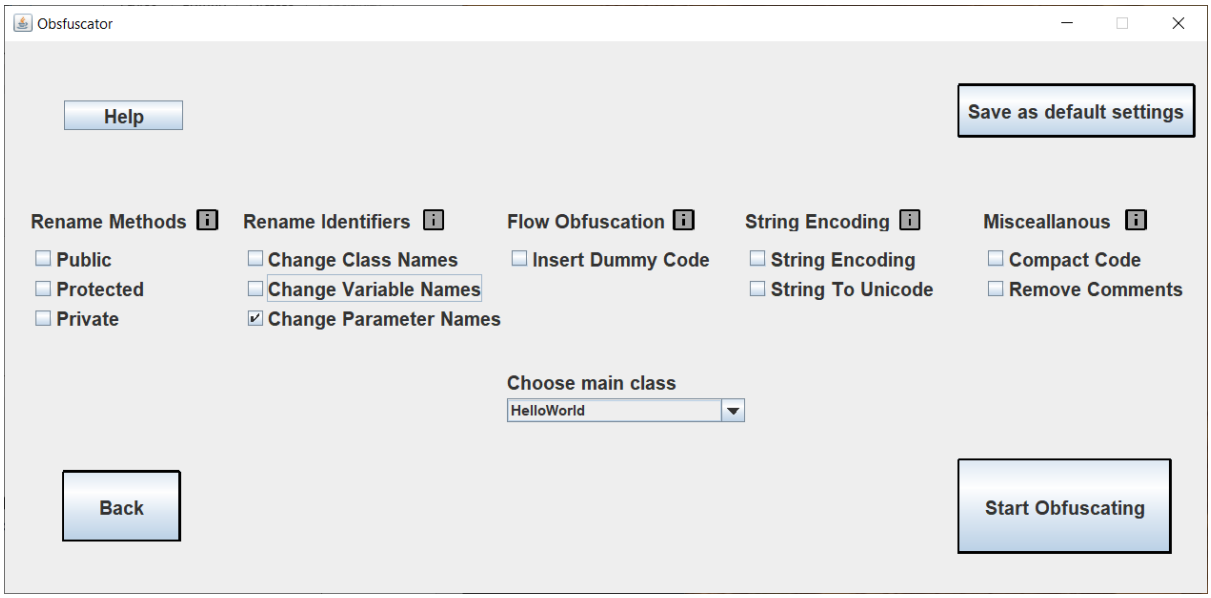
Using the technique of obfuscation, the variable identifiers will have a change of name. Referring to this example below, you can see that the text ‘test,test2,test3’ is being obfuscated. All ‘test,test2,test3’ is being changed to undistinguishable texts.

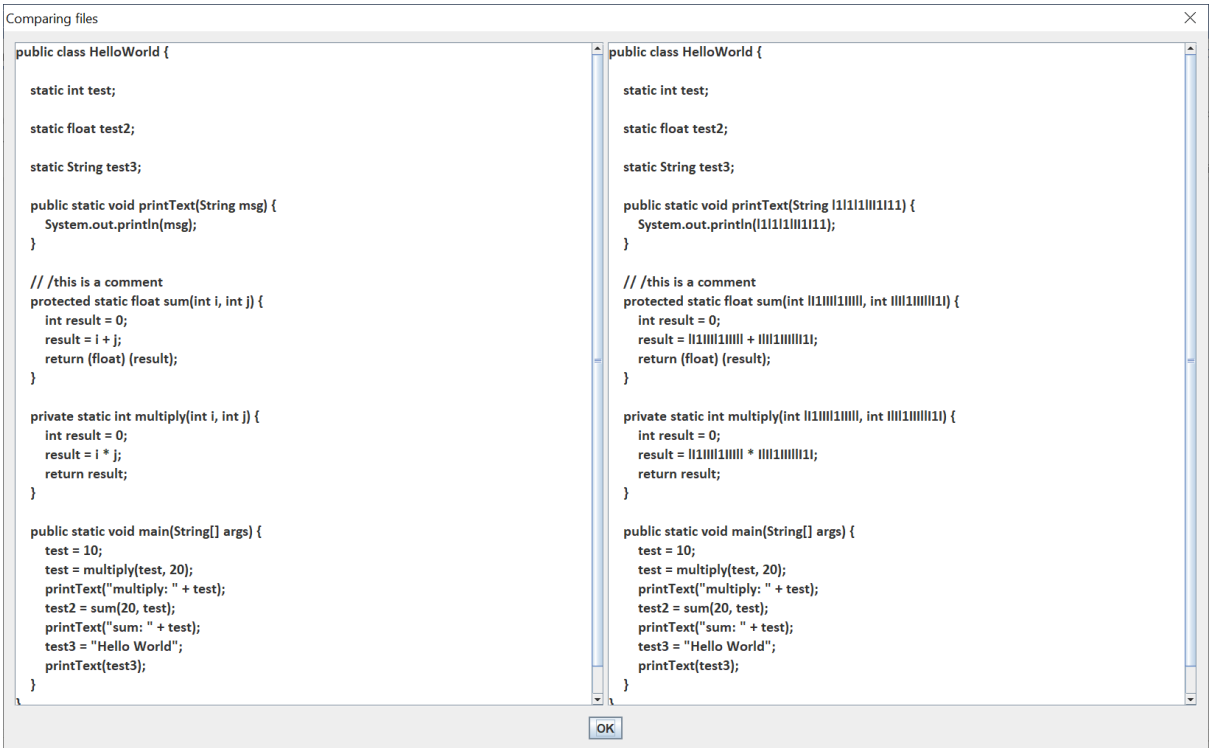




2.6 Change Parameter Names

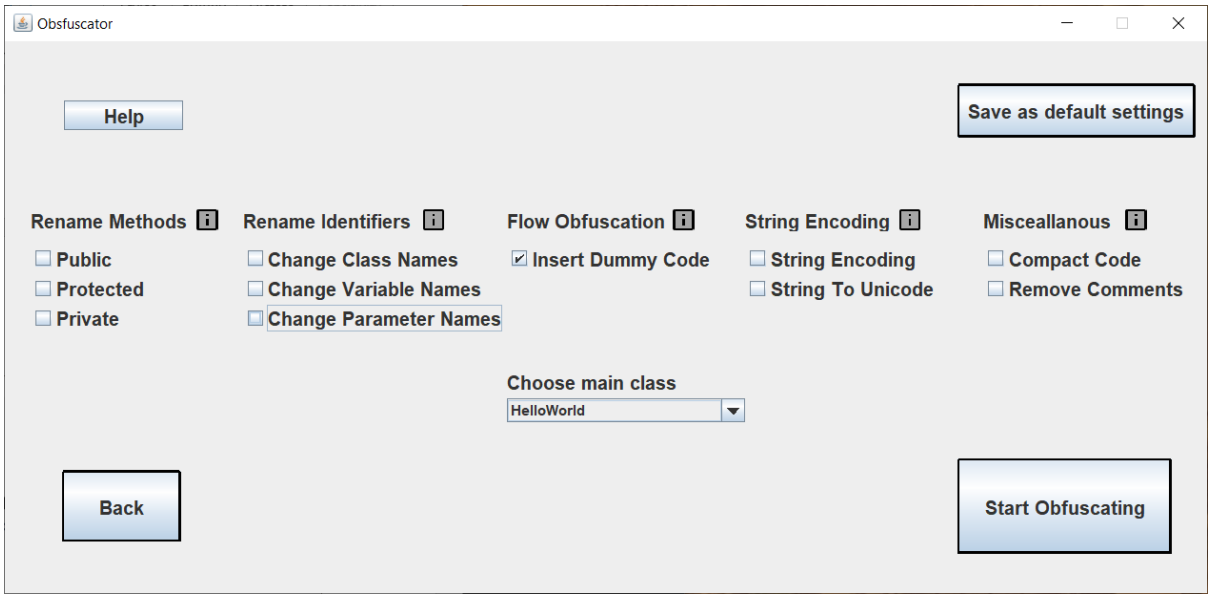
Using the technique of obfuscation, the parameter identifiers will have a change of name. Referring to this example below, you can see that the text ‘msg,I,j’ is being obfuscated. All ‘msg,I,j’ is being changed to undistinguishable texts.

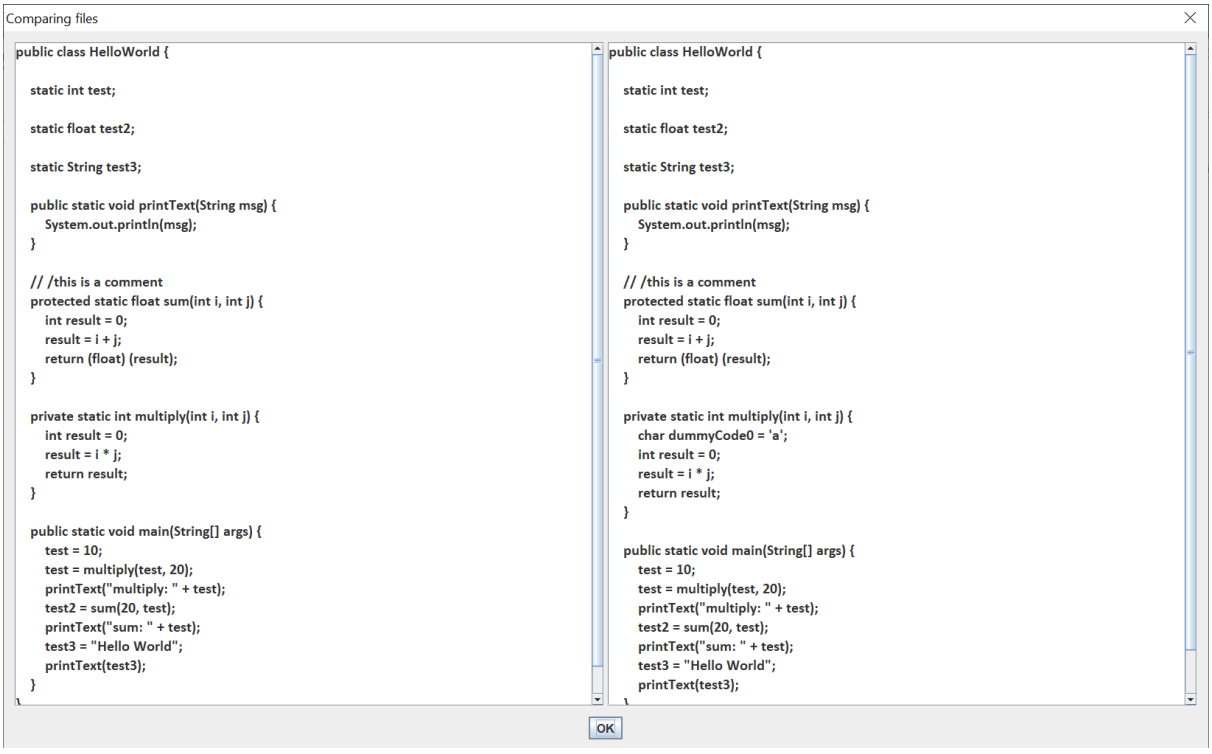




2.7 Insert Dummy Code

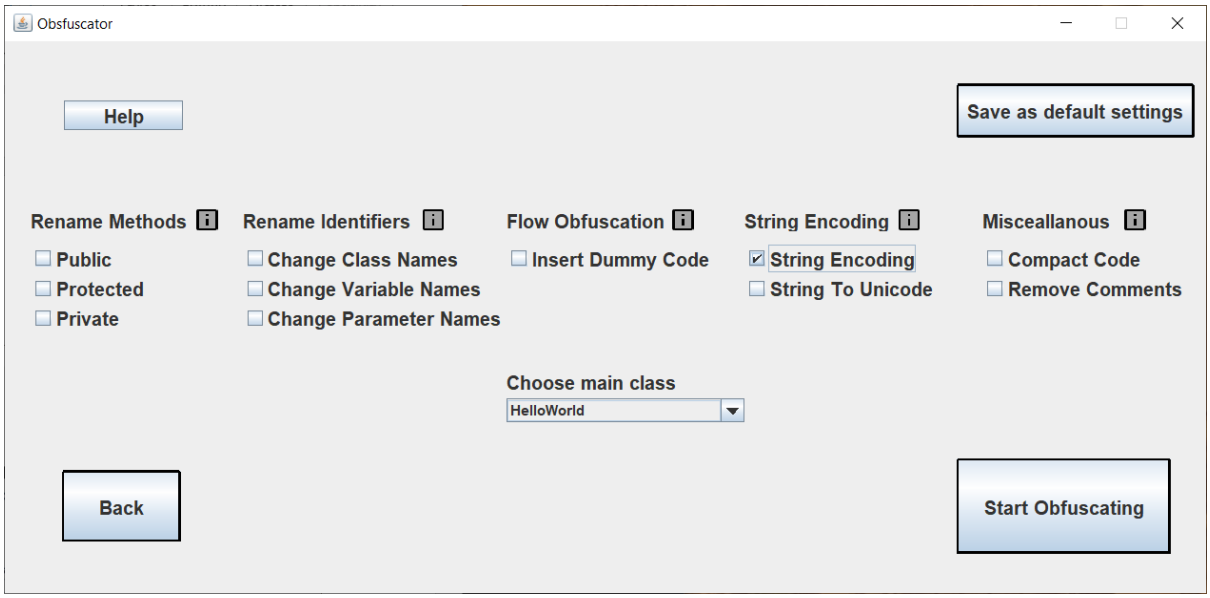
Using the technique of obfuscation, the flow will have a change. Redundant codes will be added. This however does not change the original execution of the program. Referring to this example below, you can see that the text ‘double dummycode0’ is being added.

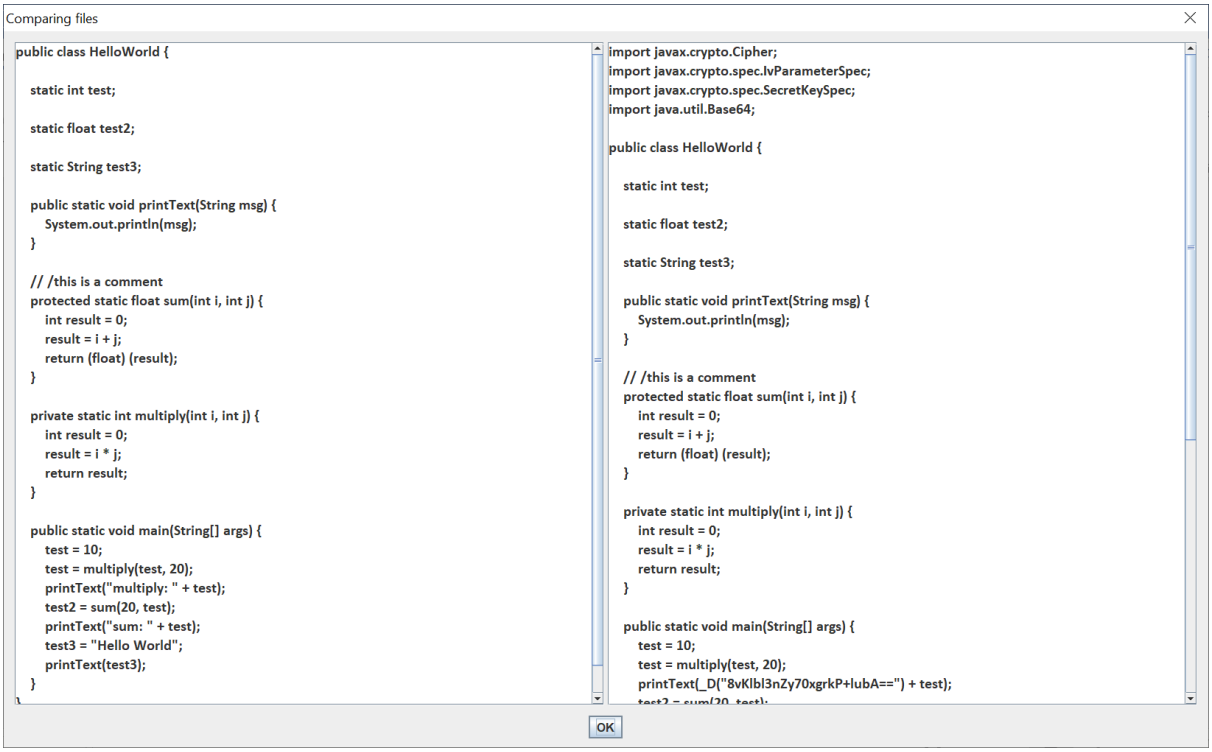


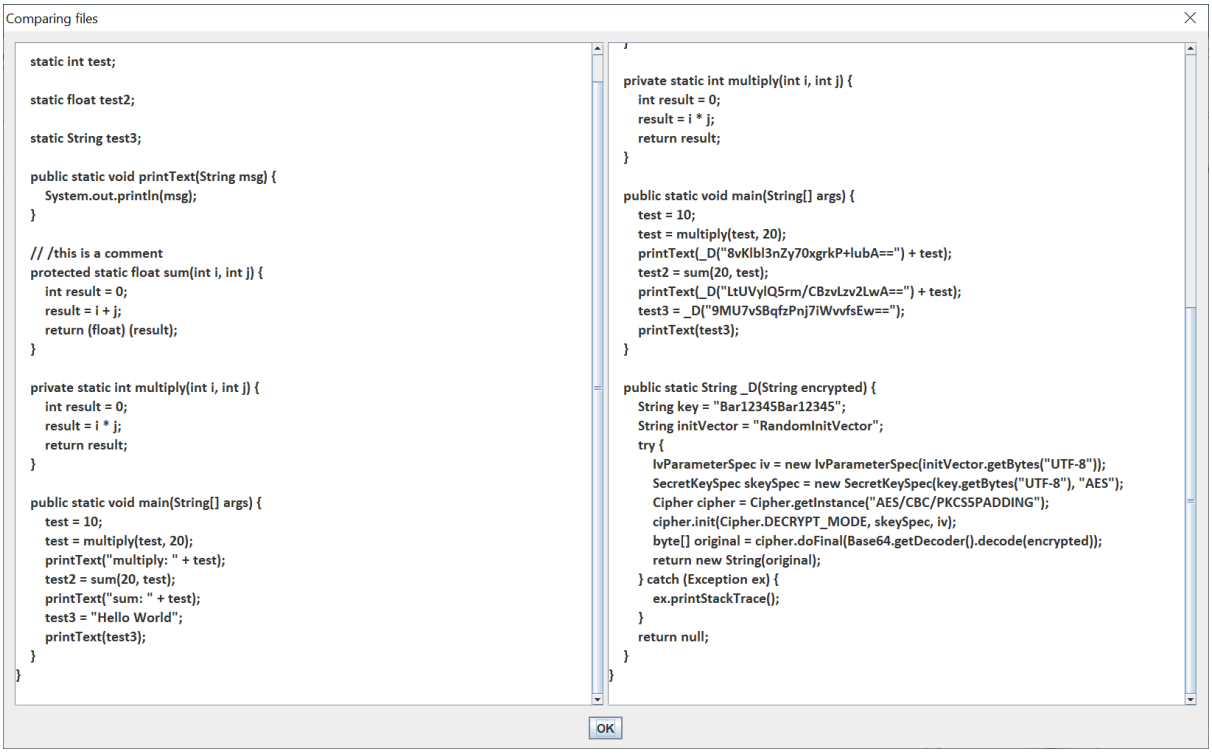


2.8 String Encoding

Using the technique of obfuscation, the stings will have a change. This however does not change the original execution of the program. Referring to this example below, you can see that the string identifiers are now change and indistinguishable from its values.

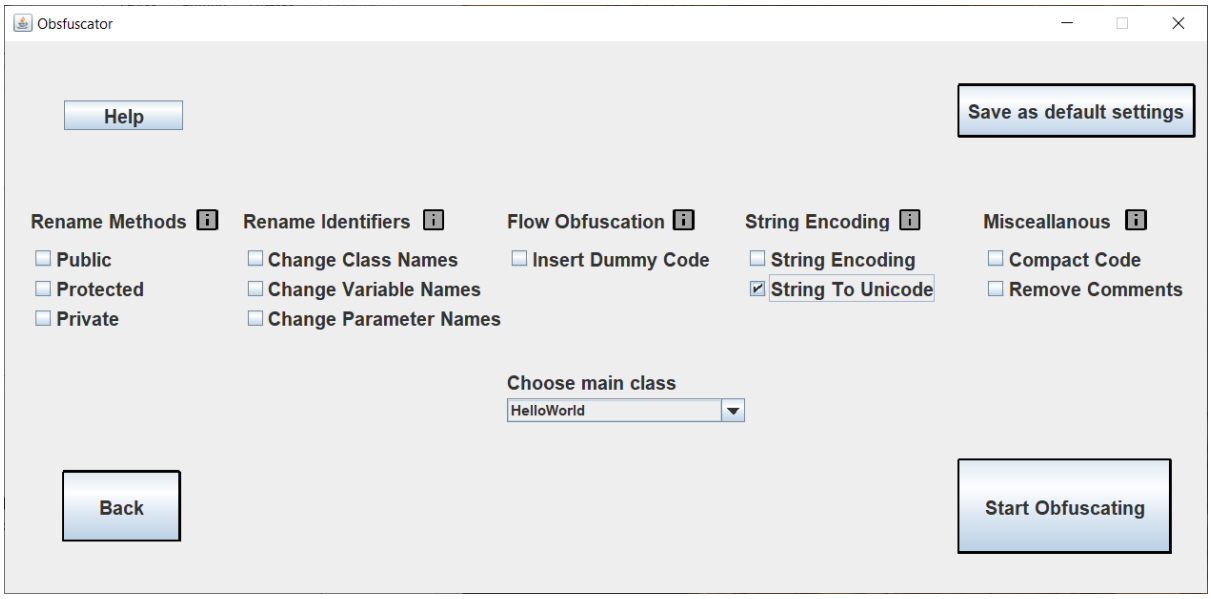


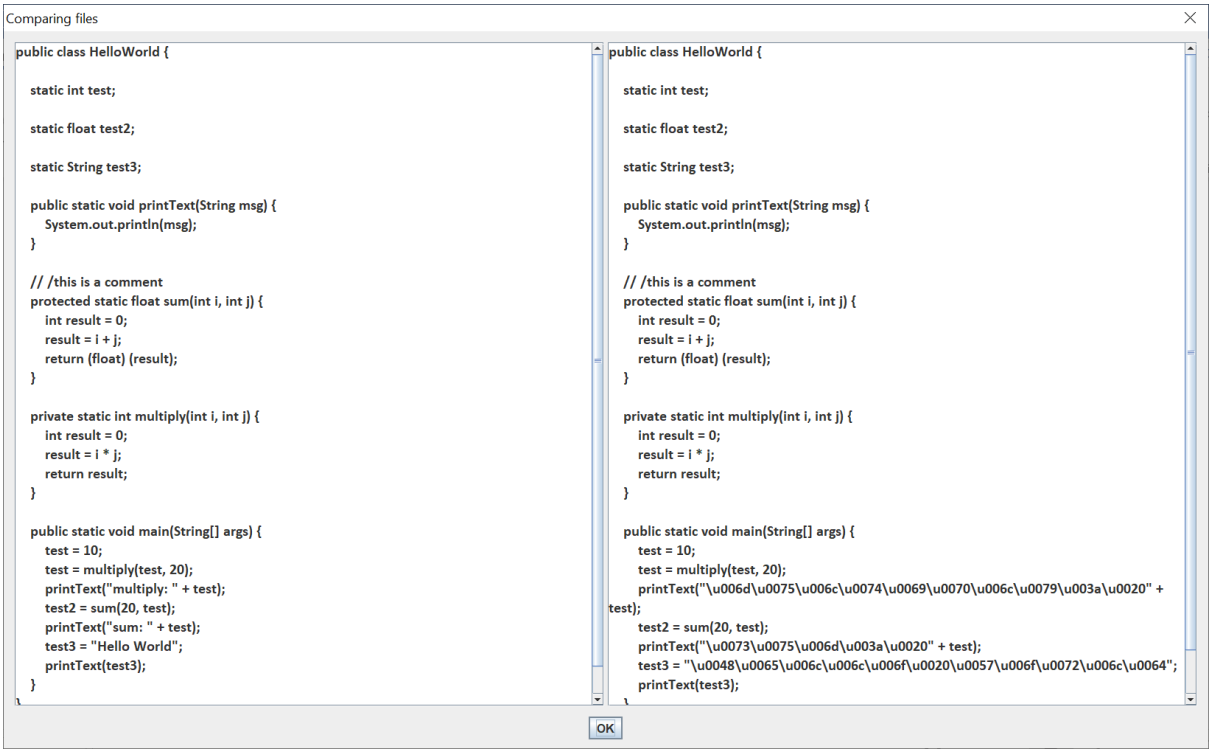

Comparing files
public class HelloWorld {
static int test;
static float test2;
static String test3;
public static void printText(String msg) {
System.out.println(msg);
}
// /this is a comment
protected static float sum(int i, int j) {
int result = 0;
result = i + j;
return (float) (result);
}
private static int multiply(int i, int j) {
int result = 0;
result = i * j;
return result;
}
public static void main(String[] args) {
test = 10;
test = multiply(test, 20);
printText("multiply: " + test);
test2 = sum(20, test);
printText("sum: " + test);
test3 = "Hello World";
printText(test3);
}
import javax.crypto.Cipher;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;
import java.util.Base64;
public class HelloWorld {
static int test;
static float test2;
static String test3;
public static void printText(String msg) {
System.out.println(msg);
}
// /this is a comment
protected static float sum(int i, int j) {
int result = 0;
result = i + j;
return (float) (result);
}
private static int multiply(int i, int j) {
int result = 0;
result = i * j;
return result;
}
public static void main(String[] args) {
test = 10;
test = multiply(test, 20);
printText(_D("8vKlbl3nZy70xgrkP+lubA==") + test);
OK


Comparing files
static int test;
static float test2;
static String test3;
public static void printText(String msg) {
System.out.println(msg);
}
// /this is a comment
protected static float sum(int i, int j) {
int result = 0;
result = i + j;
return (float) (result);
}
private static int multiply(int i, int j) {
int result = 0;
result = i * j;
return result;
}
public static void main(String[] args) {
test = 10;
test = multiply(test, 20);
printText("multiply: " + test);
test2 = sum(20, test);
printText("sum: " + test);
test3 = "Hello World";
printText(test3);
}
}
}
private static int multiply(int i, int j) {
int result = 0;
result = i * j;
return result;
}
public static void main(String[] args) {
test = 10;
test = multiply(test, 20);
printText(_D("8vKlbl3nZy70xgrkP+lubA==") + test);
test2 = sum(20, test);
printText(_D("LtUVylQ5rm/CBzvLzv2LwA==") + test);
test3 = _D("9MU7vSBqfzPnj7iWvvfsEw==");
printText(test3);
}
public static String _D(String encrypted) {
String key = "Bar12345Bar12345";
String initVector = "RandomInitVector";
try {
IvParameterSpec iv = new IvParameterSpec(initVector.getBytes("UTF-8"));
SecretKeySpec skeySpec = new SecretKeySpec(key.getBytes("UTF-8"), "AES");
Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5PADDING");
cipher.init(Cipher.DECRYPT_MODE, skeySpec, iv);
byte[] original = cipher.doFinal(Base64.getDecoder().decode(encrypted));
return new String(original);
} catch (Exception ex) {
ex.printStackTrace();
}
return null;
}
}
OK

2.9 String to Unicode

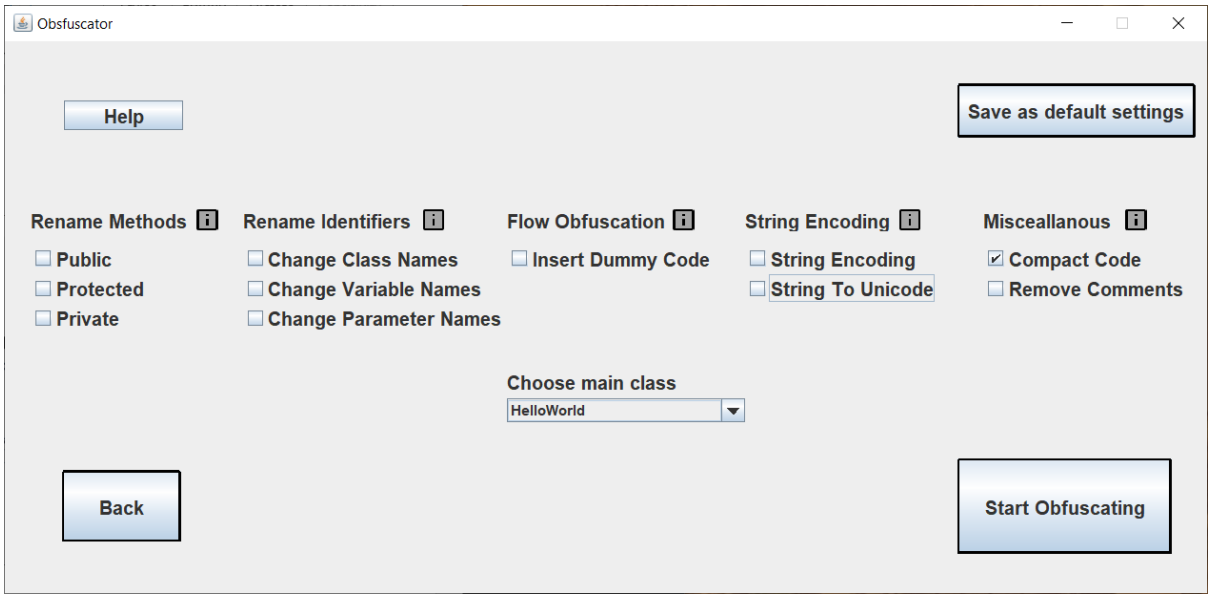
Using the technique of obfuscation, the stings will have a change. This however does not change the original execution of the program. Referring to this example below, you can see that the string identifiers are now change and indistinguishable from its values.

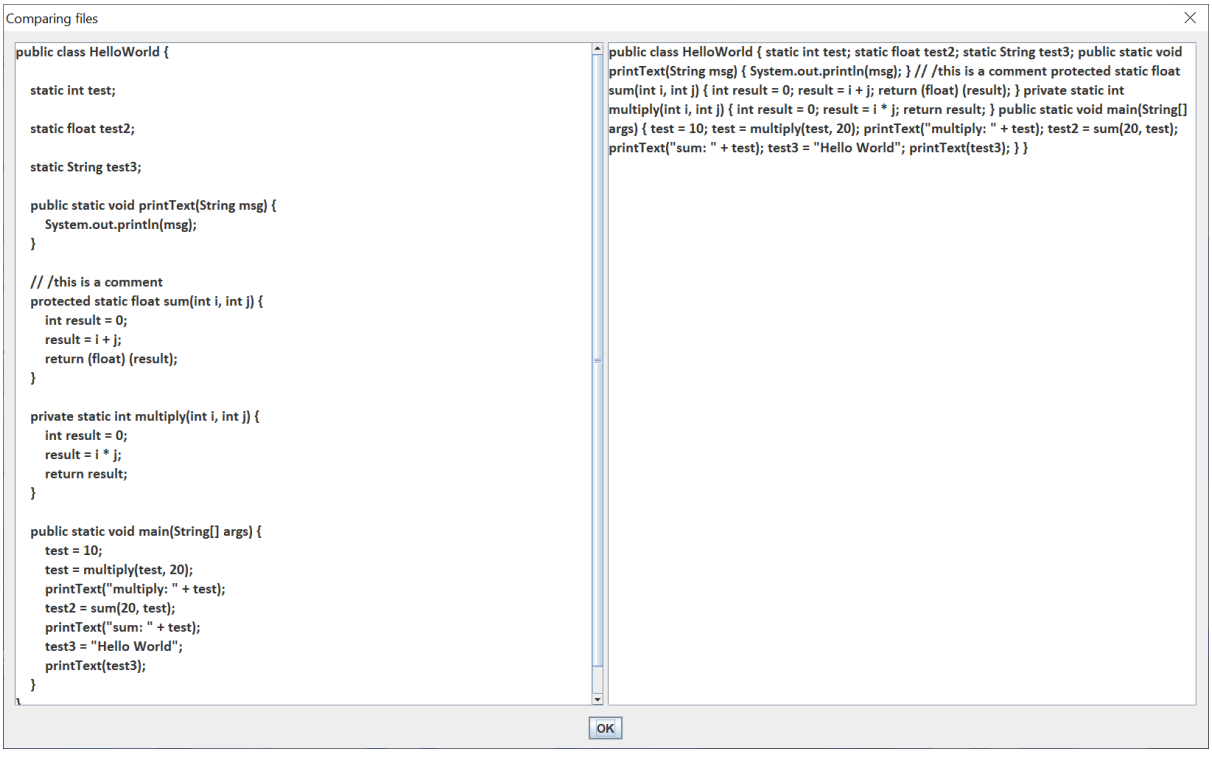




2.10 Compact Code

Using the technique of obfuscation, the code outlook will have a change. This however does not change the original execution of the program. Referring to this example below, you can see that all the white spaces are removed to compact the code. This makes it harder for reading it.





2.11 Remove Comments

Using the technique of obfuscation, the comments will be removed. This however does not change the original execution of the program. Referring to this example below, you can see that the comments starting with // and /* are now removed.