The Republic of Iraq

Ministry of Higher Education and Scientific Research

University of Babylon

College of Science for Girls

Department of Computer Science

Text Plagiarism detection

A Project Submitted to the council of College of Science for woman,
University of Babylon in Partial Fulfillment of the Requirement
For Degree of Bachelor of Science in Computer Science

Submitted by the student:

Shaima Muhammad Hamza

Supervised by:

Assistant Prof. Dr. Ali Yakoob Al-Sultan

2024

"رَبِّ أَوْزِعْنِي أَنْ أَشْكُرَ نِعْمَتَكَ الَّتِي أَنْعَمْتَ عَلَيَّ وَعَلَىٰ وَالِدَيَّ وَأَنْ أَعْمَلَ صَالِحًا تَرْضَاهُ وَأَصْلِحْ لِي فِي ذُرِّيَّتِي ۖ إِنِّي تُبْتُ إِلَيْكَ وَإِنِّي مِنَ الْمُسْلِمِينَ"

سورة الاحقاف :الايه:15 .

إلى من كلل العرق جبينه . . وشققت الأيام يديه

إلى من علمني أن الأعمال الكبيرة لا تتم إلا بالصبر والعزيمة والإصرار

أبي الغالي

إلى النور الذي ينير لي درب النجاح

قطرة في بحرها العظيم . . حباً وطاعة وبرا

أمي الغالية

إلى بسمة حياتي ووردتي الجميلة ابنتي تالا

إلى من هم أقرب إلي من مروحي و استمد عنيبتي واصراي

إخواني وأخواتي الأعزاء

شكر وتقدير

أمّا بعد : فإني أحمدُ الله جلّ وعلا على ما آتاني فضله، فقد هيأ لي كل الظروف ويسر لي إنجاز هذا العمل بفضله العظيم وكرمه العميم، فله الحمد أولاً وآخراً على كل شيء سبحانه وتعالى، ثمّ أشكر أولئك الأفاضل الكرام الذين مدوا لي أيدي المساعدة خلال هذه الفترة وهم جميع الأساتذة والدكاترة والمعيدين الذين تواصلت معهم، وفي مقدمتهم الأستاذ الفاضل المشرف على رسالة البحث الأستاذ الدكتور: (علي يعقوب السلطان)، الذي لم يدّخر جهداً في تقديم المساعدة وكنت أجلس عنده لساعات طويلة أقرأ عليه وأتعلم منه ولا يتضايق من ذلك بل يسعده مساعدة كل طالب، وكان دائمًا يحثني على البحث والنجاح، ويرغّبني في ذلك ويقوّي عزيمتي عليه، فله من الله الأجر والثواب العظيم ومني كل تقدير وشكر وامتنان، حفظه الله ومتّعه بالصحة والعافية ونفع الجميع بعلمه وعطائه، كما أشكر عميدة الكلية الدكتوره: (عبير فوزي الربيعي)، وفقهما الله تعالى لكل خير لما يبذلانه من اهتمام بطلاب الكلية على مختلف المستويات، والحمد لله رب العالم

List of Contents

Chapter One:

Chapter two:

Chapter three:

Chapter four:

Chapter five:

Chapter One

General Introduction

1 .1 Introduction:

Appropriating the ideas of others without documenting them as their primary source is a form of scientific plagiarism, which some researchers view as an extinguishing phenomenon. Its advantages over its disadvantages.

The term plagiarism can be defined as copying creative written works and professional academic research Attributing it to oneself.

Scientific plagiarism appears in quoting written material or an idea belonging to an individual and publishing it as the property of the person who quoted it. Scientific plagiarism is considered one of the types of literary and scientific theft that is widely spread in the field of scientific research. Given that scientific plagiarism is one of the major common problems in the field of scientific research, It was necessary to highlight this problem to increase the level of awareness and awareness of its seriousness in order to control it and prevent it from worsening further.

Plagiarism can be referred to as a form of fraud, theft, and forgery that includes distorting the ideas or works of another person and attributing ownership of them to oneself. Scientific plagiarism is based on misusing the ideas and works of others and attributing them to oneself, which leads to harming the scientific integrity in academic research.

Plagiarism can also be defined as the behavior related to appropriating or copying written creative and artistic works and attributing ownership of them to oneself, whether partially or completely, without indicating the source or primary author of the work. Plagiarism is considered illegal and unethical copying of the works of others while violating property rights. to the actions and ideas of others.

The phenomenon of scientific plagiarism is considered one of the major problems widespread in the field of scientific research, which appears in the inappropriate use of ideas and phrases related to others' research and attributing ownership of them to oneself by not documenting them with the source and primary reference for them. Scientific plagiarism occurs due to the lack of understanding and awareness among researchers of the rules and research skills in the field

1.2 Problem Statement:

Plagiarism can be defined as "the use or close imitation of the language and thoughts of another author and the representation of them as one's own original work [1].

Many students make (intentionally or unintentionally) some type of cheating and plagiarism in their assignments. Usually, it is difficult for teachers to detect plagiarism in student assignments by hand. The detection process becomes easier, faster and more efficient if it's performed automatically (i.e., via a computerized system).

The plagiarism detection "is the processes of locating instance of plagiarism within a document wither its text or code"[2]. . Detection can be either manual or computer-assisted. Manual detection requires effort, and it is impractical in cases where many documents to be compared.

Modular Object-Oriented Dynamic Learning Environment (Moodle) is an open source Course Management System (CMS) , it's used by educators as a dynamic web site for their students; it's best tool to manage learning. Moodle can support large deployments and hundreds of thousands of students, so it can be used for schools. It can be used to conduct fully online courses while others used it to augment face-to-face courses. Some teachers use Moodle to deliver material to students, assignments and quizzes, that makes learning more richly and collaborative.

1.3 Project Objectives:

The main goal of the project is to create an online plagiarism detection system that can be used by teachers at Palestine Polytechnic University (PPU) to detect cheating and plagiarism cases in student submitted assignments. The system should be integrated with the e-learning platform that is currently used at PPU (namely: Moodle).

The project objective mentioned in Section 1.2 can be achieved by specifying the following goals:

1. Studying, analyzing, and comparing existing open-source plagiarism detection software and algorithms.
2. Picking one of the analyzed algorithms to be used in this system, or designing a new one from the scratch. This goal includes coding the algorithm using PHP (the programming language used in Moodle).
3. Integrating the plagiarism-detection code in the existing PPU e-learning platform (Moodle).

Chapter Two

2.1 Introduction:

The phenomenon of scientific plagiarism is considered one of the major problems widespread in the field of scientific research, which appears in the inappropriate use of ideas and phrases related to others' research and attributing ownership of them to oneself by not documenting them with the source and primary reference for them. Scientific plagiarism occurs due to the lack of understanding and awareness among researchers of the rules and research skills in the field. Academic, so the issue of academic plagiarism is one of the problems that represents a great danger to researchers within academic communities.

2.2 Reasons for plagiarism:

First: The clear lack of understanding:

Some students unintentionally commit behaviors related to academic plagiarism due to a lack of awareness of the appropriate methods of quoting, paraphrasing, documenting, and arranging references in scientific research.

Second: Increasing efficiency:

Some researchers plagiarize to obtain higher grades and save time, as the researcher believes that he gets the best results based on effort, not achievement.

Third: Time management:

Some researchers face problems with time management due to the pressures of social life, commitment to university activities, family responsibilities, and completing multiple work tasks in short times, which leads to attempts at plagiarism to complete research work.

Fourth: Self-values and behaviors:

Some researchers view plagiarism as a cause of social pressure, as some view plagiarism as something acceptable and indicative of intelligence in quoting

Fifth: Research Challenge:

Some researchers view plagiarism as a concrete way to show defiance and disrespect for authorities.

Sixth: Lack of deterrent effects:

Some researchers consider that the advantages of plagiarism outweigh its risks, especially with the belief that plagiarism can go undetected or get punished for plagiarism.

2.3 Types of plagiarism

1. Direct plagiarism: which includes complete or partial copying of text, computer files, or audio and video recordings without indicating their original author.
2. Indirect plagiarism: which is based on quoting ideas and opinions from original sources while writing a few words and phrases without documenting them with their original source.
3. Accidental plagiarism: which occurs due to deficiencies in scientific skills related to academic research, which appears in not documenting ideas and phrases from their original references, formulating some words that may appear to be the researcher's own, as well as not using the foundations and rules related to documenting ideas and works attributed to others.
4. Self-Plagiarism: which is based on reusing one's own work without mentioning the main author of the idea or work.

2.4 Avoid text plagiarism:

1. Paraphrasing: When the researcher obtains useful information for the research, he reads and writes it in his own style.
2. Quoting: It is considered one of the most effective methods to avoid plagiarism by placing phrases and ideas quoted from authors between quotation marks.
3. Citation or quote: The citation is marked with a number at the end of the citation, mentioning the original reference from which the text was quoted.
4. Documenting a private work: If the author uses a citation at the beginning of the written work, the citation must be documented with the original reference mentioned to avoid plagiarism.
5. Listing references: A list of references must be placed at the end of the research, which includes the sources that the researcher used to extract the ideas and information presented in his research.
6. Follow the rules: The researcher must follow the rules in documenting references in an appropriate manner while documenting ideas taken from conferences and informal and formal conversations.
7. References include complete information about the sources used by the researcher.
8. Identify all sources used in the research in the reference list.
9. Use quotation marks if more than 6 consecutive words are copied from the original text.

2.5 Plagiarism detection systems:

It is software that compares the entered or suspicious document with related sources for the purpose of determining locations and levels of similarity in preparation for detecting and evaluating plagiarism. Many of these systems are language-specific systems, and of course they are external systems.

1. Plagiarism detection systems work using different linguistic or writing units. Some of these systems are based on letters or on words.
2. Some other systems work using syntactic or semantic features. The system analyzes the natural language of the document in two domains: parts of speech and semantic parameters.

Regardless of the linguistic or written unit used by these different systems, there are two famous methods by which the process of detecting overt plagiarism is managed:

- The first method: The plagiarism detection system performs a comparison of selected samples of the document being verified with other sources available to the system.

- The second method: The system uses key lexical units to examine similarity.

Chapter Three

3.1 Introduction:

The plagiarism checking app using Python is a software application designed to help users check for plagiarism in any document or text file they upload to the system. It is particularly useful for academic institutions, research firms, and writers who want to ensure that their work is original and free from any plagiarism. To achieve this, the application uses a combination of different technologies and libraries, including Selenium, BeautifulSoup and Docx. Selenium is used to automate web browsers, while BeautifulSoup is used to parse HTML code and extract relevant information. Docx is used to convert the document to a compatible format for comparison with the Turnitin plagiarism checker. The Turnitin plagiarism checker is a widely accepted and recognized tool that is commonly used in academic institutions and research organizations. By using this tool, the plagiarism checking app can provide a detailed plagiarism percentage report that indicates the level of similarity between the uploaded document and previously published works. One of the advantages of the plagiarism checking app using Python is that it provides a simple and user-friendly interface. This makes it easy for users to check plagiarism without the need for any technical skills or expertise. The app also provides detailed reports highlighting plagiarized content, which can help users identify and remove the offending content quickly.

3.2 Explanation of the operational code

```
import tkinter as tk
from tkinter import filedialog
from difflib import SequenceMatcher

def load_file_or_display_contents(entry, text_widget):
    file_path = entry.get()
    if not file_path:
        file_path = filedialog.askopenfilename()
    if file_path:
        entry.delete(0, tk.END)
        entry.insert(tk.END, file_path)
        with open(file_path, 'r') as file:
            text = file.read()
           text_widget.delete(1.0, tk.END)
            text_widget.insert(tk.END, text)
```

Figure (1)

Step1: To build a plagiarism detector, you will use the Tkinter modules. Tkinter is a simple, cross-platform library that you can use to create graphical user interfaces quickly.

Step2:. The Difflib module is part of the Python standard library, which provides classes and functions that compare strings such as strings, lists, and files. Thanks to it, you can build programs like autocorrect text, simple version control system or a text summarization tool.

Step3: Define a method load_file_or_display_contents() that takes entry and text_widget as arguments. This method will load a text file and display its content in a text widget

Step4: Use get() to get the file path. If the user does not enter any information, use askopenfilename() to open a file dialog window to select the file you want to check

for plagiarism. If the user selects this file path, deletes the previous entry, if any, from start to finish and inserts the selected path

Step5: Open the file in read mode and save the content in the text variable . Delete the content of text_widget and insert the text you retrieved earlier.

```
def compare_text(text1, text2):
    d = SequenceMatcher(None, text1, text2)
    similarity_ratio = d.ratio()
    similarity_percentage = int(similarity_ratio * 100)

    diff = list(d.get_opcodes())
    return similarity_percentage, diff
```

Figure (2)

Step6: Define a method, compare_text() that you will use to compare two pieces of text and calculate the percentage similarity. Use Difflib's SequenceMatcher() class to compare strings and determine similarities. Set the custom comparison function to None to use the default comparison and pass the text you want to compare

Use scaling to determine similarity in a floating-point format that you can use to calculate percentage similarity. Use get_opcodes() to retrieve a group of operations that you can use to highlight similar sections of text and return that section along with the percentage of similarity.

```
def show_similarity():
    text1 = text_textbox1.get(1.0, tk.END)
    text2 = text_textbox2.get(1.0, tk.END)
    similarity_percentage, diff = compare_text(text1, text2)
    text_textbox_diff.delete(1.0, tk.END)
    text_textbox_diff.insert(tk.END, f"Similarity: {similarity_percentage}%")
    text_textbox1.tag_remove("same", "1.0", tk.END)
    text_textbox2.tag_remove("same", "1.0", tk.END)
```

Figure (3)

Step7: Define a show_similarity() method . Use get() to retrieve the text from both text boxes and feed them to the compare_text() function . Delete the content of the resulting textbox and insert the percentage of similarity. Remove the ' same ' tag from the previous highlight (if any).

```
for opcode in diff:
        tag = opcode[0]
        start1 = opcode[1]
        end1 = opcode[2]
        start2 = opcode[3]
        end2 = opcode[4]

        if tag == "equal":
            text_textbox1.tag_add("same", f"1.0+{start1}c", f"1.0+{end1}c")
            text_textbox2.tag_add("same", f"1.0+{start2}c", f"1.0+{end2}c")
```

Figure (4)

Step8: get_opcode() returns 5 tuples: opcode string, first string start index, first string end index, second string start index, and second string end index.The opcode string can be one of four values: replace, delete, insert, and equal. You would use replace when part of the text in both strings is different, and someone has replaced part of the content with another. Delete will be used when part of the text exists in the first string, not the second.Insert is used when part of the text is not present in the first string but in the second string. You get equal results when the pieces of content are

the same. Store all these values in the appropriate variables. If the opcode string is equal , add the same tag to the text string.

```
root = tk.Tk()
root.title("Text Comparison Tool")
frame = tk.Frame(root)
frame.pack(padx=10, pady=10)

text_label1 = tk.Label(frame, text="Text 1:")
text_label1.grid(row=0, column=0, padx=5, pady=5)
text_textbox1 = tk.Text(frame, wrap=tk.WORD, width=40, height=10)
text_textbox1.grid(row=0, column=1, padx=5, pady=5)
text_label2 = tk.Label(frame, text="Text 2:")
text_label2.grid(row=0, column=2, padx=5, pady=5)
text_textbox2 = tk.Text(frame, wrap=tk.WORD, width=40, height=10)
text_textbox2.grid(row=0, column=3, padx=5, pady=5)
```

Figure (5)

Step9: Initialize the Tkinter root window. Name the window and define a frame within it. Arrange the frame with appropriate padding in both directions. Define two labels to show Text 1 and Text 2 . Set the parent component it's inside and what it displays Define 3 text boxes, two for the text you want to compare and one to show the results. Declare the parent element, width and height, set the packing option to tk.WORD to ensure that the program wraps words at the nearest boundary and doesn't break any words in between.

```
file_entry1 = tk.Entry(frame, width=50)
file_entry1.grid(row=1, column=2, columnspan=2, padx=5, pady=5)
load_button1 = tk.Button(frame, text="Load File 1", command=lambda:
load_file_or_display_contents(file_entry1, text_textbox1))
load_button1.grid(row=1, column=0, padx=5, pady=5, columnspan=2)
file_entry2 = tk.Entry(frame, width=50)
file_entry2.grid(row=2, column=2, columnspan=2, padx=5, pady=5)
load_button2 = tk.Button(frame, text="Load File 2", command=lambda:
load_file_or_display_contents(file_entry2, text_textbox2))
```

```
load_button2.grid(row=2, column=0, padx=5, pady=5, columnspan=2)
compare_button = tk.Button(root, text="Compare", command=show_similarity)
compare_button.pack(pady=5)

text_textbox_diff = tk.Text(root, wrap=tk.WORD, width=80, height=1)
text_textbox_diff.pack(padx=10, pady=10)
```

Figure (6)

Step10: Define 3 buttons, two to download files and one to compare. Specifies the parent element, the text it will display and the function it will run when it is clicked. Create two input widgets to enter the file path and define the parent element and its width.

Arrange all these elements in rows and columns using the grid manager. Use pack to sort compare_button & text_textbox_diff . Add the appropriate padding at the required position

```
text_textbox1.tag_configure("same", foreground="red", background="lightyellow")
text_textbox2.tag_configure("same", foreground="red", background="lightyellow")

root.mainloop()
```

Figure (7)

Step11: Highlight text has been highlighted the same on yellow background and red font color

The mainloop() function tells Python to loop through the Tkinter event and listen for the event until you close the window.

Chapter Four

Example results of plagiarism detection tool:

When running this program, it shows a window. When the Load File 1 button is pressed, a file dialog box opens and asks you to select the file. When selecting a file, this program displays the contents of the first text box. When entering the path and clicking Load File 2 , the program displays the content in the second text box. When you click the Compare button, you will have 100% similarity and it will highlight all the same text exactly.

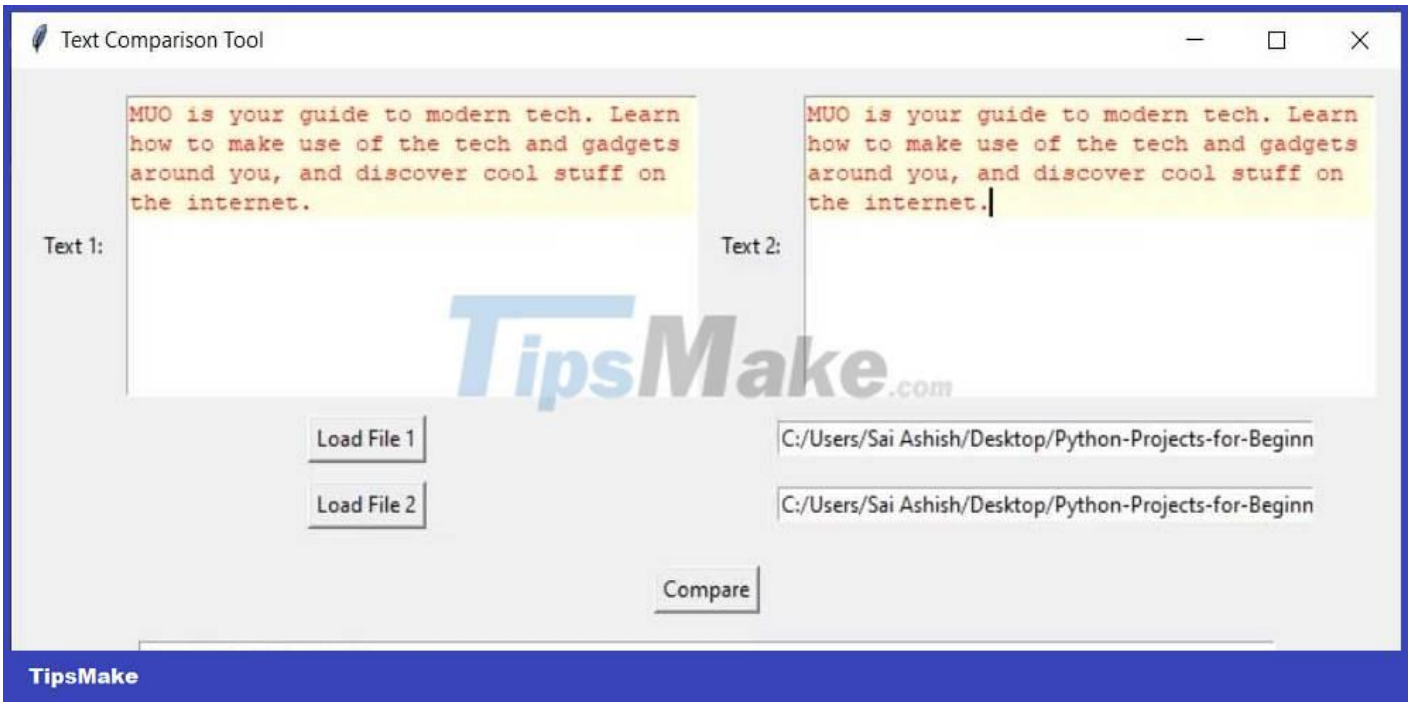


Figure (8)

If there are very few similarities, the program highlights some letters or words, but the percentage of similarity is quite low.

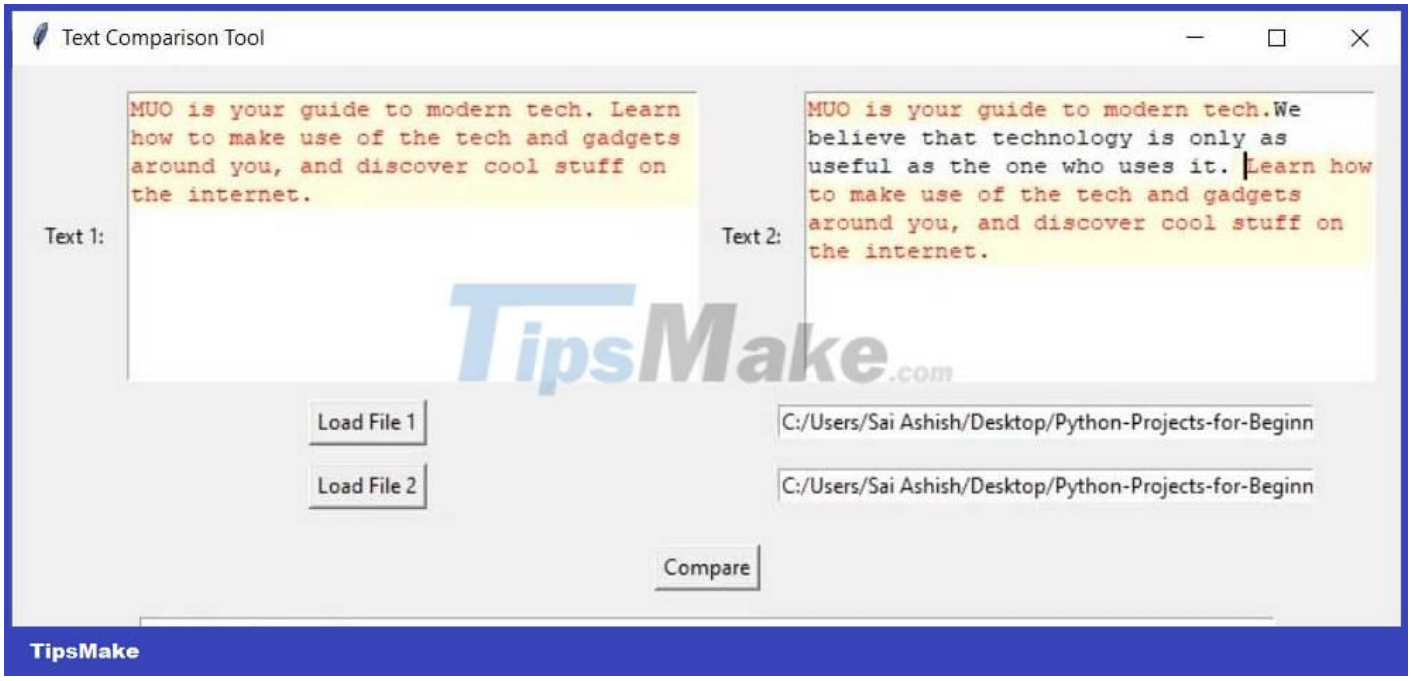


Figure (9)

If there are very few similarities, the program highlights some letters or words, but the percentage of similarity is quite low.

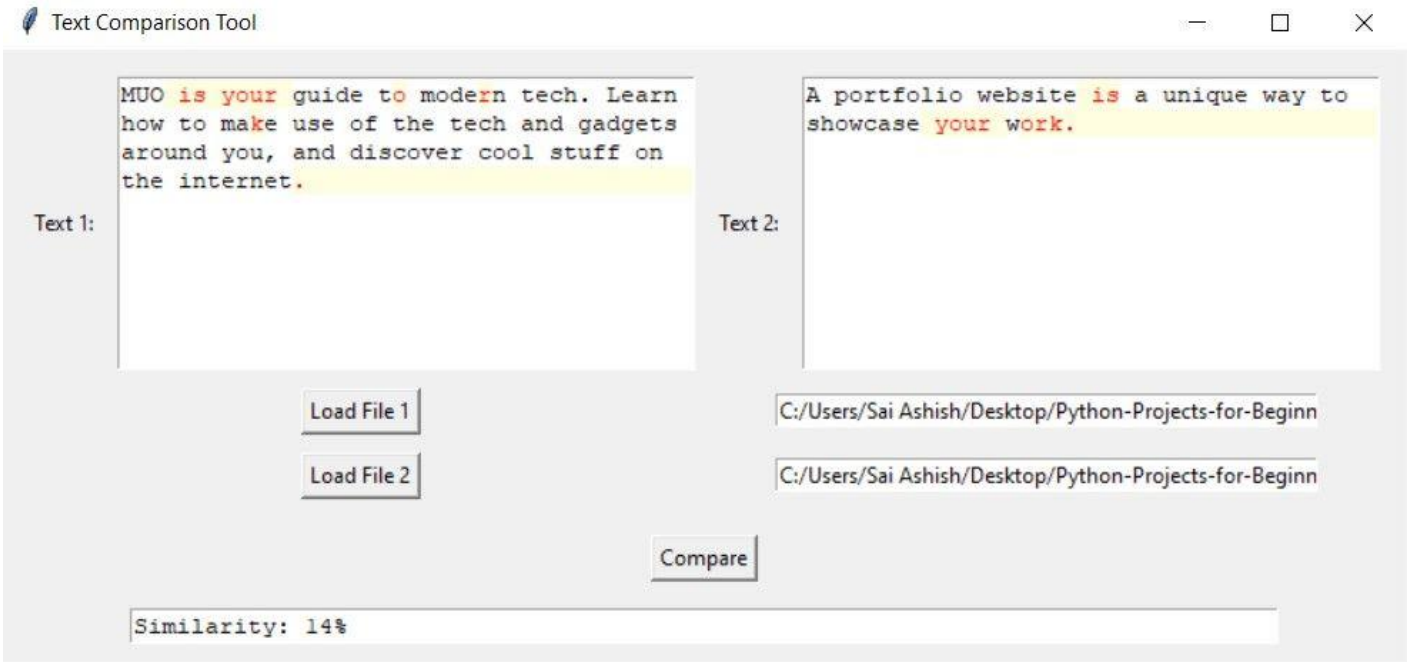


Figure (10)

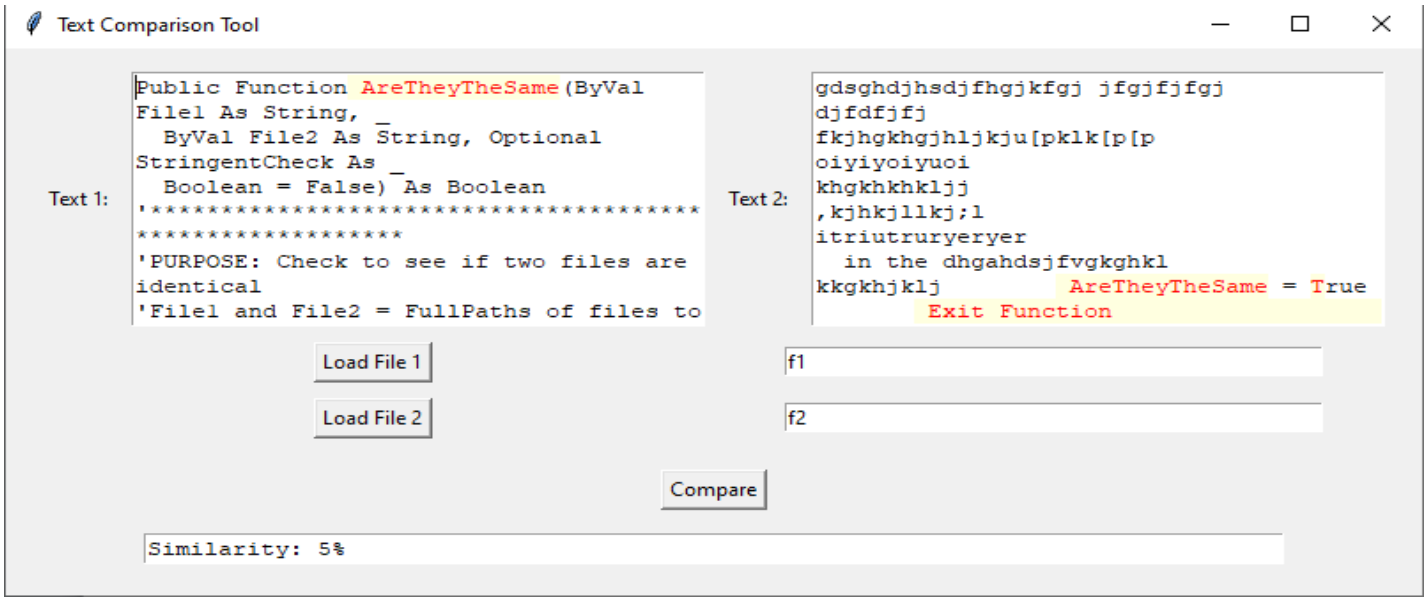


Figure (10)

****

******

def copieFichier (source, destination): "copie intégrale d'un fichier"

open(source, 'r') fd = open(destination, "w")

while 1:

txt fs.read(50). fd.write(txt)

if txt break

fs.close()

fd return

---

def filtre(source, destination):

"recopier un fichier en eliminant les lignes de renarques

Ts open(source, 'r')

fd = open(destination, "w") while 1:

EXE fs.readline() if txt=

break

if txt[0] = ''; fd.write(txt) ()

fs.close() fd.close return

****

******

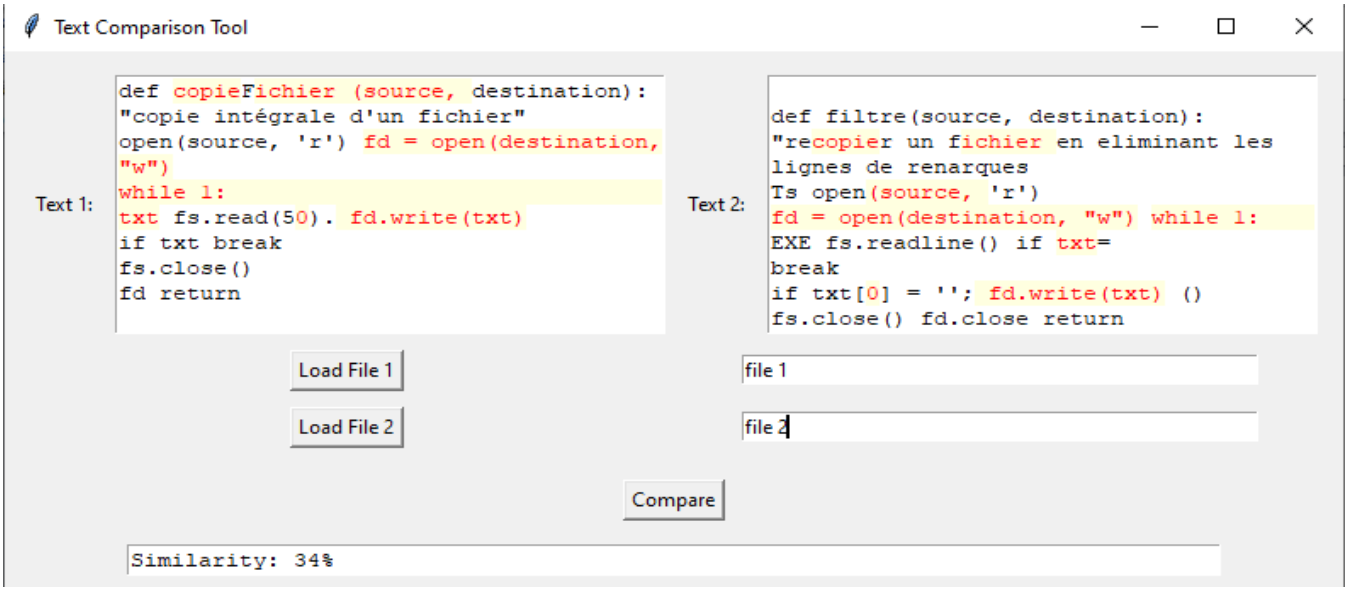


Figure (11)

Ex1: Print the sum of numbers from 1 to 100 in Python:

- First, we need a variable to store the sum value of numbers. Let it be called sum, for example, and its initial value is 0

- We repeat a loop with a range of 1 to 101 in order to calculate the number 100

Every time the loop is executed, the value of the number is added to the variable sum that we defined at the beginning Print the value of the variable sum

(Program name print 1)

```
#عرف متغير يبدأ بالصفر
sum = 0
#100 قم بعمل تكرار من 0 وحتى 101، وذلك لان أخر قيمة ستكون
for i in range (0,101):
        sum = sum + i #قم بجمع كل قيمة جديدة مع ما سبق
#أطبع القيمة النهائية لحاصل الجمع
print(sum)

#الخرج
>>> 5050
```

Ex: **Python Program to Find the Sum of Natural Numbers:**

(Program name print 2).

- To test the program for a different number, change the value of num.
- Initially, the sum is initialized to 0. And, the number is stored in variable num.
- Then, we used the while loop to iterate until num becomes zero. In each iteration of the loop, we have added the num to sum and the value of num is decreased by 1.

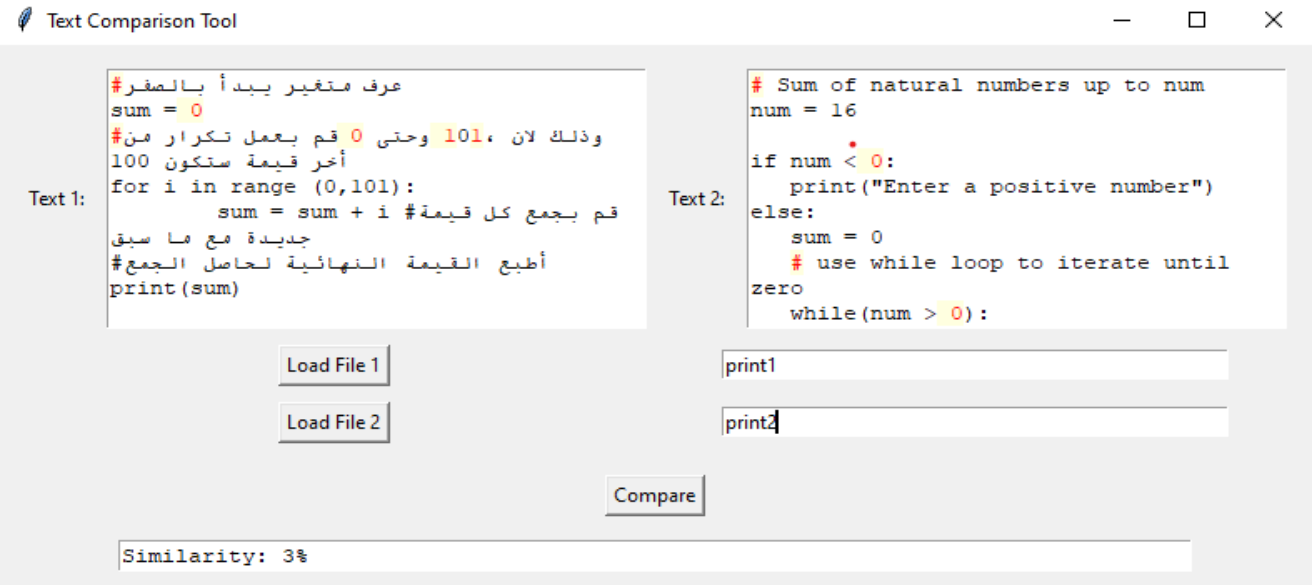
Sum of natural numbers up to num

```
num = 16

if num < 0:
   print("Enter a positive number")
else:
   sum = 0
   # use while loop to iterate until zero
   while(num > 0):
       sum += num
       num -= 1
   print("The sum is", sum)
```

Output:

```
The sum is 136
```



Ex3: In the example, we read a number from the user, send it to the function, the function divides it by 2 and returns the integer part of it. If the value is 1 or higher, the statement will be printed, but if it is 0, the if block will not be executed.

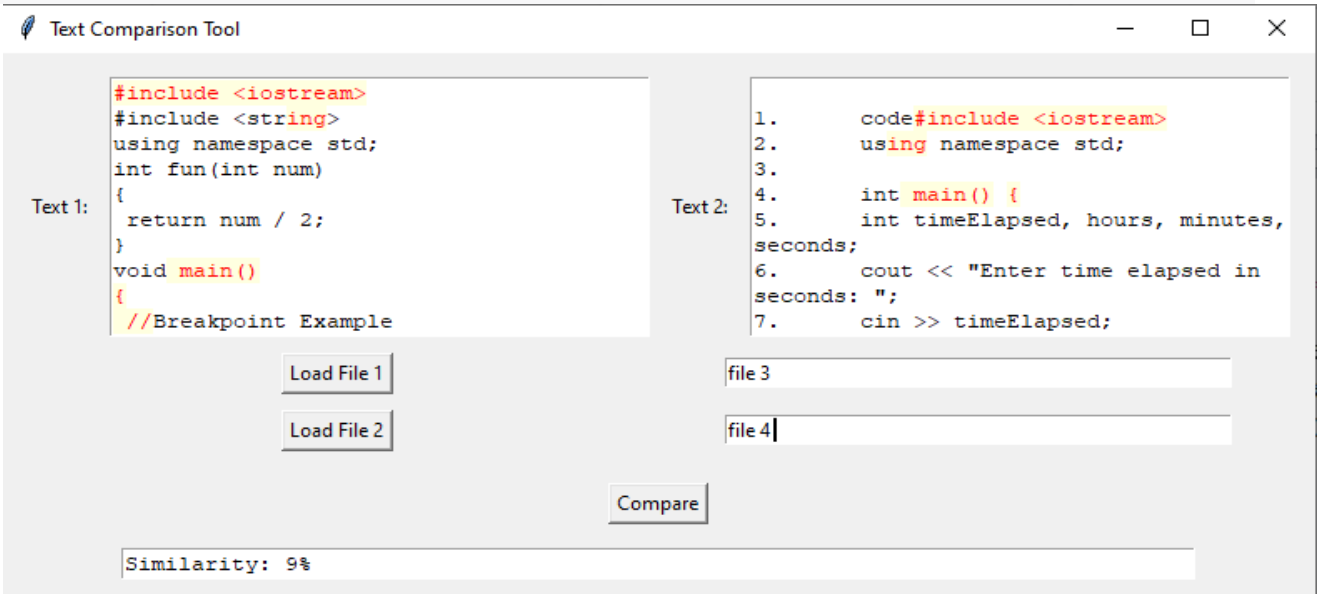
****

```
#include <iostream>
#include <string>
using namespace std;
int fun(int num)
{
 return num / 2;
}
void main()
{
 //Breakpoint Example
 int num;
 cout << " Enter num: ";
 cin >> num;
 int Array[5];
 for (int i = 0; i < 5; i++)
 Array[i] = num *i;
 if (fun(num))
 {
 cout << "The number is greater than or equal 2" << endl;
 }
 system("pause");
}
```

Ex4: Write a program in C++ that asks the user to enter the elapsed time for an event in seconds, and then the program outputs the elapsed time in hours, minutes, and seconds? (For example, if the elapsed time is (9630) seconds, the output will be (40:30):

```
1. code#include <iostream>
2. using namespace std;
3.
4. int main() {
5. int timeElapsed, hours, minutes, seconds;
```

```
6. cout << "Enter time elapsed in seconds: ";
7. cin >> timeElapsed;
8.
9. hours = timeElapsed / 3600; // الحساب على أساس 60 ثانية في الدقيقة و 60 دقيقة في الساعة
10.minutes = (timeElapsed % 3600) / 60;
11.seconds = (timeElapsed % 3600) % 60;
12.
13.cout << "Time elapsed: " << hours << ":" << minutes << ":" << seconds << endl;
14.return 0
15.}
```



Name of the first program	Name of the second program	Similarity percentage %
F1	F2	Similarity: 5%
Print1	Print 2	Similarity: 3%
File 3	File 4	Similarity: 9%

Table(1)

Chapter five

5.1 Conclusion:

1. In conclusion, the Text Plagiarism Detection project represents a significant advancement in the field of textual analysis and academic integrity. Through the utilization of advanced natural language processing techniques, machine learning algorithms, and pattern recognition methodologies, the project has successfully addressed the pressing issue of plagiarism in written content.
2. Throughout the development and implementation phases, it became evident that text plagiarism is a multifaceted problem requiring a comprehensive solution. By leveraging techniques such as cosine similarity, n-gram analysis, and semantic similarity measures, the project has demonstrated the capability to identify instances of plagiarism with a high degree of accuracy and efficiency.
3. Moreover, the project has underscored the importance of fostering a culture of academic integrity and ethical writing practices. By providing educators, researchers, and content creators with robust plagiarism detection tools, we empower them to uphold the principles of honesty, originality, and scholarly rigor in their work.
4. Looking ahead, the Text Plagiarism Detection project presents opportunities for further refinement and expansion. Future iterations could explore the integration of more sophisticated machine learning models, the incorporation of contextual information, and the development of real-time detection capabilities. Additionally, collaboration with educational institutions, publishing houses, and online platforms can facilitate widespread adoption and utilization of the tool, thereby fostering a global community committed to combating plagiarism.

5.2 Suggestion for Future Works

1. Enhanced Contextual Analysis: Future iterations of the Text Plagiarism Detection project could explore the integration of contextual analysis techniques to improve the accuracy of plagiarism detection. By considering factors such as author intent, writing style, and document genre, the system can better differentiate between legitimate paraphrasing and instances of intentional plagiarism.
2. Real-Time Detection Capabilities: Implementing real-time detection capabilities would be a valuable addition to the project, allowing users to identify instances of plagiarism as they occur. By integrating with writing platforms, academic databases, and online repositories, the system could provide immediate feedback to authors, educators, and publishers, facilitating proactive intervention and promoting a culture of originality.

5.3 References

[1] Sanjay Goel, Deepak Rao et al.: Plagiarism and its Detection in Programming Languages, December 15, 2005.

[2] Sanjay Goel, Deepak Rao et al.: Plagiarism and its Detection in Programming Languages, December 15, 2005.

[3] Ekbal, A., Saha, S., & Choudhary, G. (2012, December). Plagiarism detection in text using vector space model. In 2012 12th international conference on hybrid intelligent systems (HIS) (pp. 366-371). IEEE.

[4]. Ekbal, A., Saha, S., & Choudhary, G. (2012, December). Plagiarism detection in text using vector space model. In 2012 12th international conference on hybrid intelligent systems (HIS) (pp. 366-371). IEEE.

[5] Kong, L. L., Han, Z. Y., Qi, H. L., & Yang, M. Y. (2019). Source retrieval model focused on aggregation for plagiarism detection. Information Sciences, 503, 336-350.

[6] Barrón-Cedeño, A., & Rosso, P. (2009). On automatic plagiarism detection based on n-grams comparison. In Advances in Information Retrieval: 31th European Conference on IR Research, ECIR 2009, Toulouse, France, April 6-9, 2009. Proceedings 31 (pp. 696-700). Springer Berlin Heidelberg.

[7] Hunt, E., Janamsetty, R., Kinares, C., Koh, C., Sanchez, A., Zhan, F., ... & Oh, P. (2019, November). Machine learning models for paraphrase identification and its applications on plagiarism detection. In 2019 IEEE International Conference on Big Knowledge (ICBK) (pp. 97-104). IEEE.

[8] Stein, B., & Zu Eissen, S. M. (2006, March). Near similarity search and plagiarism analysis. In From Data and Information Analysis to Knowledge Engineering: Proceedings of the 29th Annual Conference of the Gesellschaft für Klassifikation eV

University of Magdeburg, March 9–11, 2005 (pp. 430-437). Berlin, Heidelberg: Springer Berlin Heidelberg.

[9] Khaled, F., & Al-Tamimi, M. S. H. (2021). Plagiarism detection methods and tools: An overview. Iraqi Journal of Science, 2771-2783.

[10] Butakov, S., & Scherbinin, V. (2009). The toolbox for local and global plagiarism detection. Computers & Education, 52(4), 781-788.

[11] Ciesielski, V., Wu, N., & Tahaghoghi, S. (2008, July). Evolving similarity functions for code plagiarism detection. In Proceedings of the 10th annual conference on Genetic and evolutionary computation (pp. 1453-1460).

[12] Zimba, O., & Gasparyan, A. (2021). Plagiarism detection and prevention: a primer for researchers. Reumatologia/Rheumatology, 59(3), 132-137.

[13] Kashkur, M., Parsutins, S., & Borisovs, A. (2010). Research into plagiarism cases and plagiarism detection methods. Rigas Tehniskas Universitates Zinatniskie Raksti, 44, 138.

[14] Jadalla, A., & Elnagar, A. (2012). A plagiarism detection system for Arabic text-based documents. In Intelligence and Security Informatics: Pacific Asia Workshop, PAISI 2012, Kuala Lumpur, Malaysia, May 29, 2012. Proceedings (pp. 145-153). Springer Berlin Heidelberg.

[15] Meyer zu Eissen, S., Stein, B., & Kulig, M. (2007). Plagiarism detection without reference collections. In Advances in Data Analysis: Proceedings of the 30 th Annual Conference of the Gesellschaft für Klassifikation eV, Freie Universität Berlin, March 8–10, 2006 (pp. 359-366). Springer Berlin Heidelberg.

[16] Shenoy, N., & Potey, M. A. (2016). Semantic similarity search model for obfuscated plagiarism detection in Marathi language using Fuzzy and Naïve Bayes approaches IOSR. Journal of Computer Engineering, 18(3), 83-88.

[17] Al Jarrah, A., Alsmadi, I., & Za'atreh, Z. (2011, May). Plagiarism Detection based on studying correlation between Author, Title, and Content. In International Conference on Information Communication System (CICS) (pp. 22-24).

[18] Hariharan, S. (2012). Automatic plagiarism detection using similarity analysis. Int. Arab J. Inf. Technol., 9(4), 322-326.

[19] Mondal, S., & Mondal, H. (2018). Google search: A simple and free tool to detect plagiarism. Indian Journal of Vascular and Endovascular Surgery, 5(4), 270-273.

[20] Leung, C. H., & Chan, Y. Y. (2007, October). A natural language processing approach to automatic plagiarism detection. In Proceedings of the 8th ACM SIGITE conference on Information technology education (pp. 213-218).

[21] Kiss, A. K. (2013). Loopholes of plagiarism detection software. Procedia-Social and Behavioral Sciences, 106, 1796-1803.