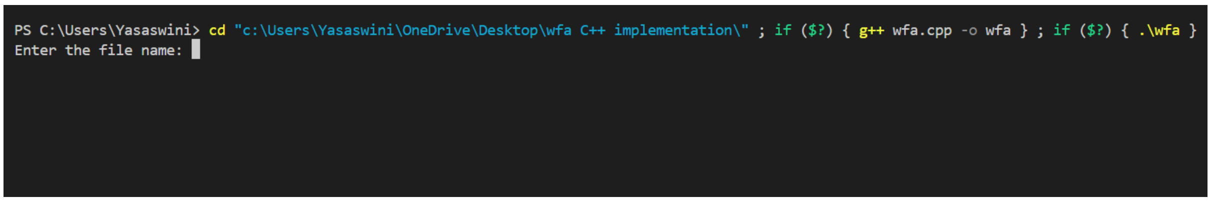
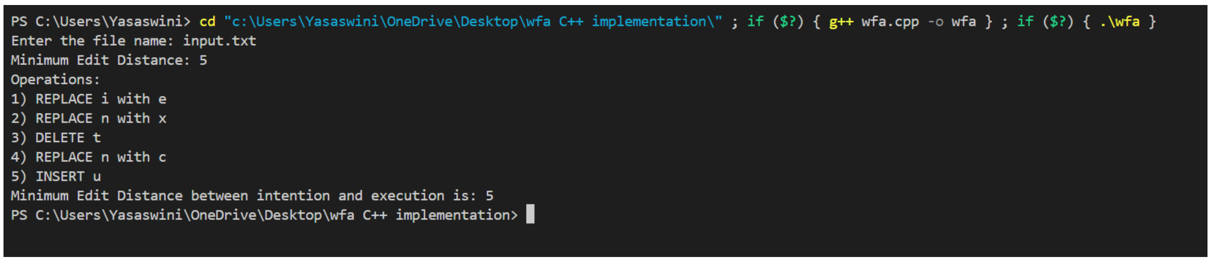
WAGNER-FISCHER ALGORITHM

Let’s examine the execution in various scenarios:

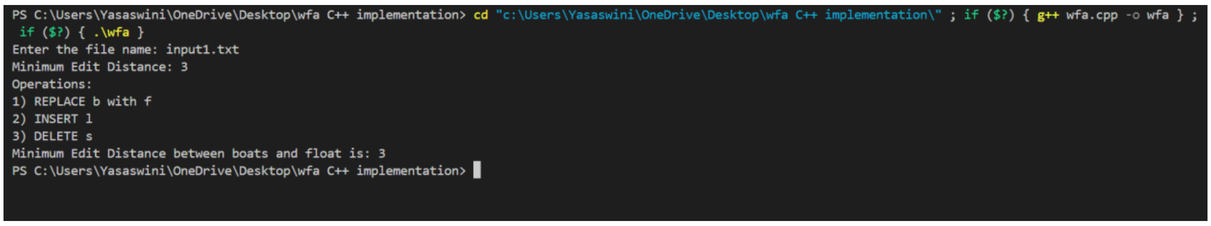
In the implementation, I have incorporated input retrieval from secondary storage. Additionally, I have included a user input prompt for entering a file name, allowing them to access different files in various program runs.



```
PS C:\Users\Yasaswini> cd "c:\Users\Yasaswini\OneDrive\Desktop\wfa C++ implementation\" ; if ($?) { g++ wfa.cpp -o wfa } ; if ($?) { .\wfa }
Enter the file name: 
```



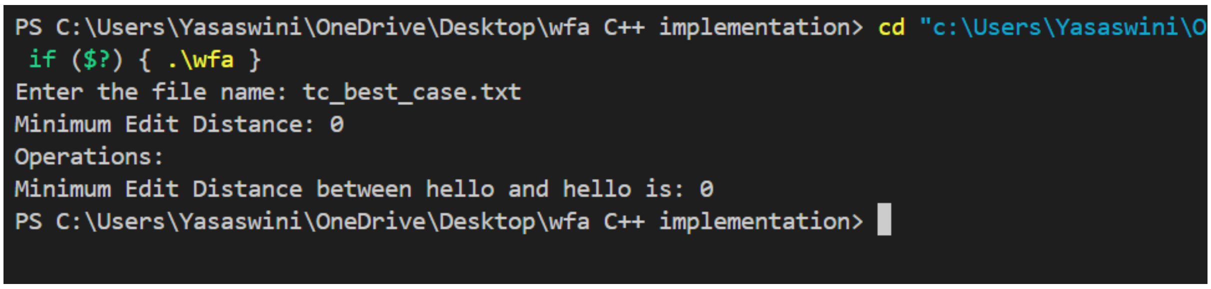
```
PS C:\Users\Yasaswini> cd "c:\Users\Yasaswini\OneDrive\Desktop\wfa C++ implementation\" ; if ($?) { g++ wfa.cpp -o wfa } ; if ($?) { .\wfa }
Enter the file name: input.txt
Minimum Edit Distance: 5
Operations:
1) REPLACE i with e
2) REPLACE n with x
3) DELETE t
4) REPLACE n with c
5) INSERT u
Minimum Edit Distance between intention and execution is: 5
PS C:\Users\Yasaswini\OneDrive\Desktop\wfa C++ implementation> 
```



```
PS C:\Users\Yasaswini\OneDrive\Desktop\wfa C++ implementation> cd "c:\Users\Yasaswini\OneDrive\Desktop\wfa C++ implementation\" ; if ($?) { g++ wfa.cpp -o wfa } ;
 if ($?) { .\wfa }
Enter the file name: input1.txt
Minimum Edit Distance: 3
Operations:
1) REPLACE b with f
2) INSERT l
3) DELETE s
Minimum Edit Distance between boats and float is: 3
PS C:\Users\Yasaswini\OneDrive\Desktop\wfa C++ implementation> 
```

BEST CASE:

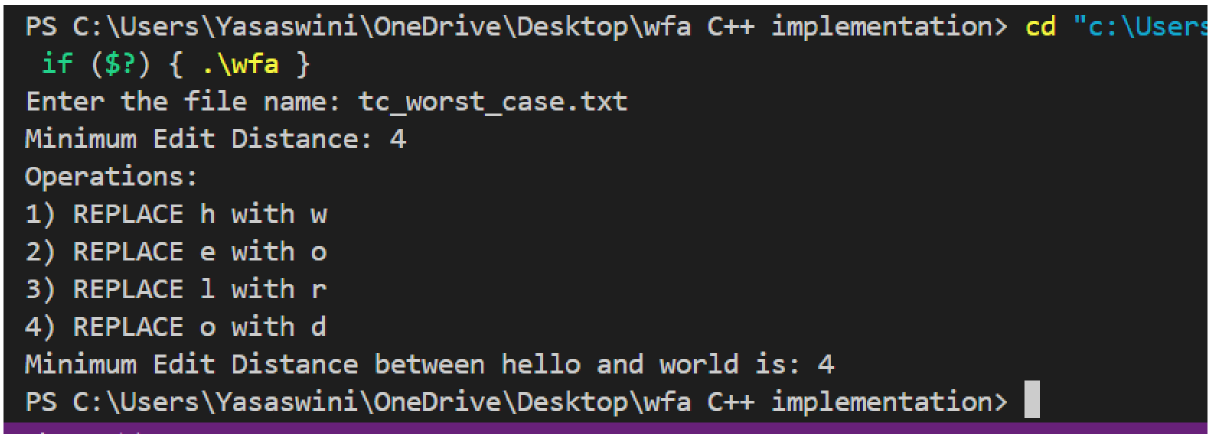
- strings are identical
- Example: str1 = "hello", str2 = "hello"
- Expected Output: Minimum Edit Distance: 0
- Explanation: Since the strings are already the same, no edits are needed, and the distance is 0.



```
PS C:\Users\Yasaswini\OneDrive\Desktop\wfa C++ implementation> cd "c:\Users\Yasaswini\O
 if ($?) { .\wfa }
Enter the file name: tc_best_case.txt
Minimum Edit Distance: 0
Operations:
Minimum Edit Distance between hello and hello is: 0
PS C:\Users\Yasaswini\OneDrive\Desktop\wfa C++ implementation>
```

WORST CASE:

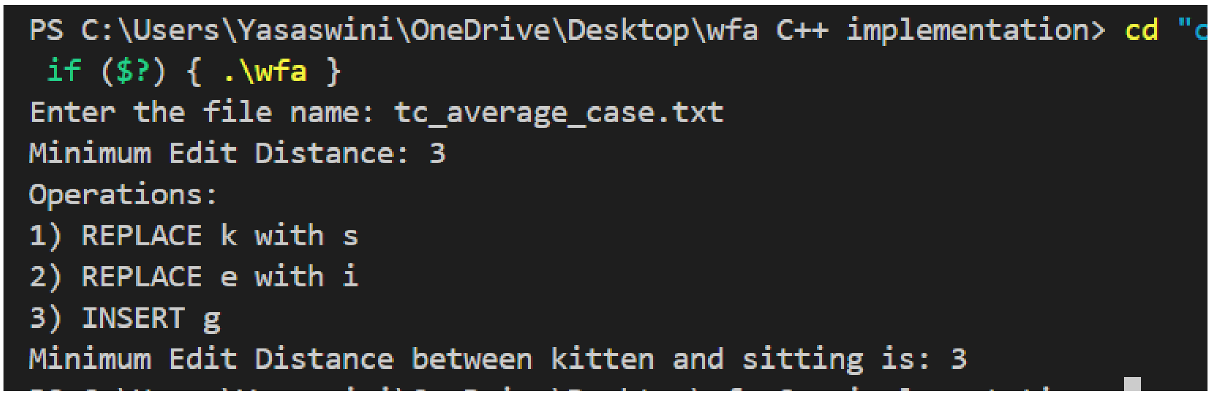
- Input strings are completely different.
- Example: str1 = "hello", str2 = "world"
- Expected Output: Minimum Edit Distance: 4
- Explanation: The algorithm needs to change each character in str1 to match str2, which requires 4 edits.



```
PS C:\Users\Yasaswini\OneDrive\Desktop\wfa C++ implementation> cd "c:\Users
 if ($?) { .\wfa }
Enter the file name: tc_worst_case.txt
Minimum Edit Distance: 4
Operations:
1) REPLACE h with w
2) REPLACE e with o
3) REPLACE l with r
4) REPLACE o with d
Minimum Edit Distance between hello and world is: 4
PS C:\Users\Yasaswini\OneDrive\Desktop\wfa C++ implementation>
```

AVERAGE CASE:

- Input strings are moderately different.
- Example: str1 = "kitten", str2 = "sitting"
- Expected Output: Minimum Edit Distance: 3
- Explanation: The algorithm needs to change 'k' to 's', 'e' to 'i', and add 'g' at the end of str1.

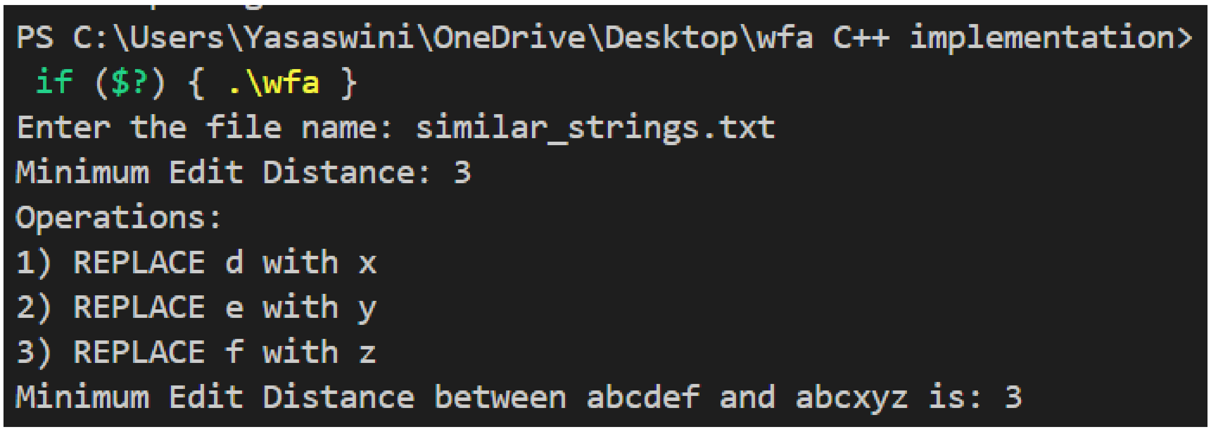


```
PS C:\Users\Yasaswini\OneDrive\Desktop\wfa C++ implementation> cd "c
 if ($?) { .\wfa }
Enter the file name: tc_average_case.txt
Minimum Edit Distance: 3
Operations:
1) REPLACE k with s
2) REPLACE e with i
3) INSERT g
Minimum Edit Distance between kitten and sitting is: 3
```

CORNER CASE:

Similarity in strings:

- Input strings have some similar prefixes but differ later.
- Example: str1 = "abcdef", str2 = "abcxyz"
- Expected Output: Minimum Edit Distance: 3
- Explanation: The algorithm needs to substitute 'd', 'e', 'f' with 'x', 'y', 'z' to match str2.



```
PS C:\Users\Yasaswini\OneDrive\Desktop\wfa C++ implementation>
 if ($?) { .\wfa }
Enter the file name: similar_strings.txt
Minimum Edit Distance: 3
Operations:
1) REPLACE d with x
2) REPLACE e with y
3) REPLACE f with z
Minimum Edit Distance between abcdef and abcxyz is: 3
```