International Journal of Artificial Intelegence Research
Vol. 7, No. 1, June 2023, pp. 39 - 55
ISSN 2579-7298

Analysis of the Similiarity Level of Source Code in the Kotlin Programming Language using Winnowing Algorithm

Yustikamasy Astica [a,1,*], Ema Utami [a,2], Anggit Dwi Hartanto [a,3]

[a] Universitas AMIKOM Yogyakarta, Jl. Ring Road Utara, Ngringin, Condongcatur, Kec. Depok, Kabupaten Sleman, Daerah Istimewa Yogyakarta, and 55281, Indonesia
[1] yustikamasy@students.amikom.ac.id *; [2] ema.u@amikom.ac.id; [3] anggit@amikom.ac.id
* corresponding author

ARTICLE INFO

Article history
Received 21 Apr 2023
Revised 07 June 2023
Accepted 27 June 2023

Keywords
Plagiarism
Winnowing algorithm
Jaccard Similarity

ABSTRACT

Plagiarism is an act of imitating the work of others directly or indirectly. In an academic environment, plagiarism applies not only to textual documents but also to source code documents. Source code plagiarism in academia usually occurs when students copy another student's code and submit it as if it were the student's work. So that an automatic plagiarism check is needed, the winnowing algorithm will be used to help detect similarities in source code as a way to detect an act of plagiarism. The Winnowing algorithm, which is usually used to detect document plagiarism, this research detects the source code. The results produced in this study are that the degree of similarity in the two source codes will produce different similarity values if the dataset used has gone through the text preprocessing stage or without preprocessing. If the dataset has gone through the text preprocessing stage, the similarity value will be pretty low because the number of characters used is significantly reduced. The Winnowing and Jaccard Similarity algorithms quickly detect plagiarism in source code and can be used to minimize plagiarism.

1. Introduction

Plagiarism is an act of imitating the work of others directly or indirectly. In the academic environment, plagiarism usually occurs in textual documents such as essays, reports, and even research. However, plagiarism does not only apply to textual records but also to source code documents. Source code plagiarism in academia usually occurs when students copy another student's code and submit it as if it were the student's work [1].

According to the Kamus Besar Bahasa Indonesia (KBBI), plagiarism or better known as plagiarism, is taking someone else's essay (opinion and so on) and making it appear as if it were your essay. In the Regulation of the Minister of National Education of the Republic of Indonesia No. 17 of 2010, article 1 defines plagiarism as an act intentionally or unintentionally in obtaining or trying to obtain credit or value for scientific work by quoting part or all of the work or scientific works of other parties that are recognized as scientific works, without citing sources accurately and adequately..

In computer science or informatics, source code is essential for programmers because source code is central to building a system or application. In making applications,

programmers have many ways to differentiate in writing program code, such as the language used, libraries, use of functions, and others. The problem that usually arises in academic circles related to computer science is the occurrence of plagiarism in programming assignments. Moreover, lately many schools or universities are using the concept of online learning, and the opportunities for plagiarism are wider because there is no direct supervision from the teacher. Thus, automation to detect plagiarism in source code that has high quality is urgently needed to meet the needs because the number of students continues to grow.

There have been many studies conducted to overcome plagiarism in source code. However, each research for plagiarism detection has different algorithms and accuracy results depending on how the researcher forms a model to solve the problems raised.

Based on [2] states that studies have shown that between 50% to 79% of undergraduate students will plagiarize at least once during their academic career. With such a high rate, an academic can assess cases of alleged plagiarism. In computer science courses, plagiarism is often encountered as source code plagiarism. Source code plagiarism occurs when a student takes someone else's source code and submits it as their work.

Several indicators are of concern to every researcher in improving the performance of plagiarism detection. According to [3] , stemming affects the accuracy of similarity values generated using stemming, resulting in fewer good scores than those without stems. So, that research can be carried out to test whether the detection uses preprocessing or without preprocessing.

Research by [4] explains that the winnowing Algorithm can detect plagiarism in a reasonably fast time. Textually, this Algorithm is very effective in dealing with copy-paste and relocation plagiarism. The selection of the k-gram value will affect the similarity value. A small k-gram value will give a more excellent similarity value. The smaller the substring, the smaller the character will affect the hash value and can provide a fingerprint value.

The value of the k and w parameters dramatically influences the resulting similarity value. Research conducted by [5] by comparing two random assignments and obtaining various similarity values. The greater the value of k and w, the smaller the resulting similarity value; the smaller the k and w value, the greater the resulting similarity value. So the value of k = 2 increases the chances of finding similarities between the two samples being compared, considering that the winnowing Algorithm works by checking each code tested.

Research by [6] found that the Winnowing Algorithm can detect plagiarism at a high level in documents, the Rabin Karp algorithm at an intermediate level and the Knuth Morris Pratt algorithm can detect plagiarism at a low level in documents that have the same subject, with the arrangement gram = 5, window = 7 and base = 3. The analysis results confirm that the optimal Algorithm of the three is the Winnowing Algorithm, which produces a similarity of 94.3%. It also shows the highest accuracy, with a difference of only 1.19% from inspection by human experts, at 93.11%.

Research conducted by [8] proposes a machine-learning approach to detect plagiarism in programming assignments. This study performs calculations based on different features such as similarity scores, the number of variables and functions that are not used, etc., which come from the source code. The XGBoost learning algorithm is used and compares the results with the Support Vector Machine (SVM). The dataset used was programming assignments submitted by students during two introductory programming courses at the University of Sarajeva, which were found online in the C/C++ programming language and were then sorted to separate pairs of plagiarism and non-plagiarism. With the created model, the results obtained an accuracy score of 94% and an average f1 score of 0.905 in

the test set, and when compared to SVM, the model using XGBoost is better in the dataset used.

Research by [5] used the winnowing algorithm to detect plagiarism in programming source code by using ten student assignments using the winnowing algorithm and obtaining various similarity values by comparing two tasks randomly. And it can be concluded that the parameter values k and w significantly affect the resulting similarity value, namely the greater the k and w value, the smaller the resulting similarity value, and the smaller the k and w value, the greater the resulting similarity value. So the value of k = 2 is used to increase the chances of finding similarities between the two samples being compared, considering that the winnowing algorithm works by checking each code tested.

The plagiarism detection research conducted by [9] used the Mamber and Winnowing algorithms on abstract text documents. The dataset used is a conceptual document from students uploaded to the internet. In its implementation, this study uses two approaches, namely the Biword and Triword approaches embedded in the winnowing algorithm, while Biword is for the Manber algorithm. The testing results with ten documents showed that the average similarity of the mamber algorithm is 90.56% while the winnowing algorithm is 94%. And for the winnowing algorithm with the triword approach produces an accuracy of 91.22%. So the biword process and the winnowing algorithm are better than the Manber algorithm and the winnowing algorithm with the triword method.

Another research conducted by [10] was motivated by the fact that manually checking source code plagiarism is a repetitive, complex, and time-consuming task, so automation is needed to detect plagiarism in source code that has high-quality. The dataset used in this study uses the Java programming language collected from Petra Christian University programming classes. This study uses three main algorithms: Levenshtein distance, greedy string tiling, and bigram, producing 12 features and statistical features. Then in the final step, features will be used to process training and inference with the XGBoost model. The test results show that using the proposed features and preprocessing has better performance metrics than previous research, namely an f1-score of 99%. The preprocessing application can also improve performance metrics based on previous studies' proposed features.

Winnowing is an algorithm based on a hashing approach that applies a hash function and window formation to obtain fingerprints when matching patterns. This algorithm is used by [11] based on words (word-level) still needs to be done, so this study aims to measure the level of similarity of dishes using the Winnowing algorithm and word-level trigrams. The results showed that the Winnowing algorithm applied using word-level trigrams could detect similarities in the text by 76.84%, 52.29%, 37.40%, and 19.29%. From the research results, the pattern-matching method with the Winnowing algorithm and word-level trigrams can be used to measure the level of text similarity.

The research conducted [12] found that the determination of the best parameters using the winnowing algorithm was based on the smallest value of the difference in parameters (differences in the results of similarity based on parameters), and the highest similarity value was the result of the similarity dice and the Jaccard coefficient. The results of setting the best research parameters with hash = 5, k-gram = 2, window = 7.The results of testing these parameters indicate that the higher the k-gram value will affect the results of the similarity value. And if the hash or window value is taller, the change in result similarity is not too large but significant enough to affect the differences between parameters. Testing the value of the similarity level using the Jaccard similarity in the winnowing algorithm is lower than the dice coefficient with the difference between the dice similarity and the Jaccard coefficient of 2.554683%.

Based on some of the research above, this study will conduct experiments in detecting plagiarism. This study uses the Winnowing Algorithm, the winnowing Algorithm was

applied by comparing the results when using the Kotlin grammar preprocessing and not preprocessing. Thus, it can produce a better model against various plagiarism attacks and performs tests using Jaccard similarity. In conducting this research, the winnowing Algorithm will be used to help see similarities in source code which will be used as a way to detect the presence of an act of plagiarism. The Jaccard similarity is required as a document fingerprint, and the Algorithm that will be used to support it is the winnowing Algorithm. This is by statement [7] Jaccard similarity is usually used to compare documents and calculate the similarity value of two objects or documents.

2. Method

This research was conducted to detect plagiarism in source code using the Winnowing Algorithm.

The research stage can be seen in Figure 1.

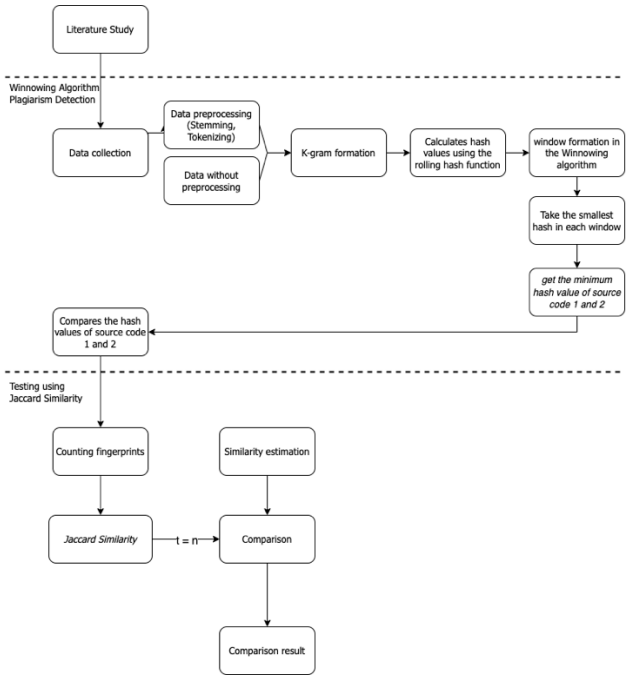

Figure 1. Proposed Work

2.1. Dataset

The dataset is obtained from GitHub with the Kotlin programming language with the same theme. Then manually select the source code that has a code that is close to plagiarism or not. Then the data will be processed using text preprocessing using stemming and tokenizing. There are two types of datasets used, there are dataset that has been processed with text preprocessing and with preprocessing. The two data are then processed with the winnowing algorithm

2.2. Pre-processing

Stemming removes parts that tend to be static and are the same for all code, such as package declarations, class declarations, and main function declarations. This method is done so that the general part does not affect the assessment of whether a source code is plagiarized. Tokenizing functions to group each token from the source code into several categories. These categories are whitespace, comments, strings, operators, keywords, functions, variables, and numbers. The tokenizer used in this study uses the Pygment library, pygment is used to highlight the syntax of particular programming languages. Inside the pygment library is a library called KotlinLexer which functions for generating tokens and parsing Kotlin code in your program using the Kotlin grammar. The grouping of token

categories is taken from the Kotlin Programming Language specification. Then we will form k-grams and calculate the value using the rolling hash.

2.3. Proposed Work

The process begins with collecting data via Github. The data collected is the source code of the Kotlin programming language with the theme "Creating a Github User App application" and taking one of the files with the .kt extension.

Then data processing is carried out, and the data is manually selected source code with a code close to plagiarism or not. Then the data will be processed using text preprocessing using stemming and tokenizing. Stemming removes parts that tend to be static and are the same for all code, such as package declarations, class declarations, and main function declarations. This is done so that the general part does not affect the assessment of whether a source code is a plagiarism—tokenizing functions group each token from the source code into several categories.

After processing the data, the data is used to detect plagiarism in the source code using the Winnowing algorithm.

Here is how the Winnowing algorithm works [13]:

1. Build a k-gram series from the text
2. Perform a hash function for each gram. Equation (1) is the calculation of the hash function of the winnowing algorithm

$$H_{(C_1 \ldots C_k)} = C_1 * b^{k-1} + C_2 * b^{k-2} + C_{k-1} * b^k + c_k$$

$$H_{(C_2 \ldots C_{k+1})} = (H_{(C_2 \ldots C_k)} - c_1 * b^{k-1}) * b + c_{k+1} \quad (1)$$

   Details:

   H = hash value

   c = characters in grams

   b = base number

   k = gram character count.
3. Creates sets called windows consisting of i hash values. If i = 6, then in one window, there are 6 hash values
4. Selecting the fingerprint from the hashing results by dividing the hash results based on one window w value, and then selecting the smallest hash value from each window.

The fingerprint results obtained by comparing the two source codes will be tested using Jaccard similarity.

2.4. Evaluation

Testing by calculating similarity using Jaccard similarity. Two experiments were conducted, namely, the fingerprint results using preprocessing and not preprocessing. The fingerprint results will be compared using the Jaccard similarity equation. The Jaccard similarity value is obtained from the intersection divided by the union of the two sets. Jaccard distance is a measure of dissimilarity between data sets. This can be determined by the inverse of the Jaccard coefficient, obtained by removing the Jaccard similarity from the Jaccard similarity value. The advantage of Jaccard similarity is that it calculates the number of terms that are the same in each sentence and compares it to the total number of terms in both sentences. Jaccard similarity can be formulated as follows [13]:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B} \quad (2)$$

Details:

J = Jaccard Similarity

A= Source code 1

B= Source code 2

3. Results and Discussion

The trial results of the plagiarism detector using Jaccard Similarity or the Jaccard Coefficient for the winnowing algorithm are shown in Table 1 using different datasets. The dataset used is the exact same source code which is the source code with the same program code, source code that is close to the same is the source code which is differentiated in the function calls and initialization of the variables, and the source code which has a big difference is the source code which has different calling activities.

The following are the stages of implementing the Winnowing algorithm which are described in the following test.

Source code 1

```
package com.submission.githubuserapplication

import android.content.Intent
import android.content.res.TypedArray
import android.os.Bundle
import android.widget.AdapterView
import android.widget.ListView
import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {

    private lateinit var adapter: UserAdapter
    private lateinit var dataUsername: Array<String>
    private lateinit var dataName: Array<String>
    private lateinit var dataCompany: Array<String>
    private lateinit var dataPhoto: TypedArray
    private lateinit var dataLocation: Array<String>
    private lateinit var dataRepository: Array<String>
    private lateinit var dataFollowers: Array<String>
    private lateinit var dataFollowing: Array<String>
    private var users = arrayListOf<User>()

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val actionbar = supportActionBar
        actionbar!!.title= "List User Github"

        val listView: ListView = findViewById(R.id.lv_list)
        adapter = UserAdapter(this)
        listView.adapter = adapter

        prepare()
        addItem()

        listView.onItemClickListener =
AdapterView.OnItemClickListener{ _, _, position, _ ->
            val selectedUser: User = users[position]
            val moveIntentWithParcelable =
Intent(this@MainActivity, DetailUser::class.java)

moveIntentWithParcelable.putExtra(DetailUser.EXTRA_USER,
selectedUser)
            startActivity(moveIntentWithParcelable)
        }
    }

    private fun prepare() {
        dataUsername =
resources.getStringArray(R.array.username)
        dataName = resources.getStringArray(R.array.name)
        dataCompany =
resources.getStringArray(R.array.company)
        dataPhoto =
resources.obtainTypedArray(R.array.avatar)
        dataLocation =
resources.getStringArray(R.array.location)
        dataRepository =
resources.getStringArray(R.array.repository)
        dataFollowers =
resources.getStringArray(R.array.followers)
        dataFollowing =
resources.getStringArray(R.array.following)
```

Source code 2

```
package com.masuwes.githubuserfirst

import android.content.res.TypedArray
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    private lateinit var dataName: Array<String>
    private lateinit var dataUserName: Array<String>
    private lateinit var dataCompany: Array<String>
    private lateinit var dataRepository: Array<String>
    private lateinit var dataFollowers: Array<String>
    private lateinit var dataFollowing: Array<String>
    private lateinit var dataAvatar: TypedArray
    private lateinit var dataLocation: Array<String>
    private var users = arrayListOf<User>()
    private lateinit var adapter: UserAdapter

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val actionBar = supportActionBar
        actionBar!!.setTitle("User List")


        val         listView:         RecyclerView         =
findViewById(R.id.listViewUser)


        adapter = UserAdapter()

        listView.adapter = adapter

        prepare()
        addData()
        showRecyclerView()

    }

    private fun prepare() {
        dataUserName                                        =
resources.getStringArray(R.array.username)
        dataName = resources.getStringArray(R.array.name)
        dataLocation                                        =
resources.getStringArray(R.array.location)
        dataRepository                                      =
resources.getStringArray(R.array.repository)
        dataCompany                                         =
resources.getStringArray(R.array.company)
        dataFollowers                                       =
resources.getStringArray(R.array.followers)
        dataFollowing                                       =
resources.getStringArray(R.array.following)
        dataAvatar                                          =
resources.obtainTypedArray(R.array.avatar)
    }

    private fun showRecyclerView() {
        val layoutManager = LinearLayoutManager(this)
```

```
    }

    private fun addItem() {
        for (position in dataName.indices) {
            val user = User(
                    dataUsername[position],
                    dataName[position],
                    dataCompany[position],
                    dataPhoto.getResourceId(position, -1),
                    dataLocation[position],
                    dataRepository[position],
                    dataFollowers[position],
                    dataFollowing[position]
            )
            users.add(user)
        }
        adapter.users = users
    }
}
```

```
        listViewUser?.setLayoutManager(layoutManager)
        listViewUser?.setHasFixedSize(true)
    }

    // this method are need to make same order like the data
    private fun addData() {
        for (position in dataName.indices) {
            val user = User(
                dataUserName[position],
                dataName[position],
                dataLocation[position],
                dataRepository[position],
                dataCompany[position],
                dataFollowers[position],
                dataFollowing[position],
                dataAvatar.getResourceId(position, -1)
            )
            users.add(user)
        }
        adapter.listUser = users
    }
```

1. *Preprocessing*

In this process stemming and tokenizing are carried out which function to eliminate common parts so that they do not affect the assessment of whether a source code is plagiarism or not. In Figure 3.1 is the source code being processed text preprocessing

```
class Preprocessor:
    @staticmethod
    def clean_text(text):
        """
        Takes in a piece of text and eliminates unnecessary features such as capitalization, trailing or leading
        whitespaces, and spaces between words. This will also remove URLs and tokenize Kotlin code.
        Parameters
        ----------
        text : str
            String that we want to clean up. In practice, this is source code.

        Returns
        -------
        cleaned_text : str
            String that is the same as the input text but with the unnecessary features removed and Kotlin code tokenized.
        """
        punctuation = ["\"", "<", ">","(", ")", "=", "{","}","[","]",":",",","_","$"]
        text = str(text)
        cleaned_text = re.sub(r"http\S+", '', text) #regex code to remove URLs
        cleaned_text = cleaned_text.lower()
        cleaned_text= cleaned_text.replace(" ", "") #remove whitespace
        cleaned_text= cleaned_text.replace("\n", "") #remove the newline character
        cleaned_text= cleaned_text.replace("\t", "") #remove the tab character
        cleaned_text= cleaned_text.replace(".","")
        for punctuation_mark in punctuation:
            cleaned_text = cleaned_text.replace(punctuation_mark, "")
        lexer = KotlinLexer()
        tokens = [token[1] for token in lexer.get_tokens(cleaned_text) if token[0] != "EOF"]
        return " ".join(tokens)
```

Figure 3.1. Source code for text preprocessing process

The results after preprocessing are as below:

Source code 1

```
packagecomsubmissiongithubuserapplicationimportandroidcontentintentimportandroidcontentrestypedarrayimportandroidosbundlei
mportandroidwidgetadapterviewimportandroidwidgetlistviewimportandroidxappcompatappappcompatactivityclassmainactivityappcom
patactivityprivatelateinitvaradapteruseradapterprivatelateinitvardatausernamearraystringprivatelateinitvardatanamearraystr
ingprivatelateinitvardatacompanyarraystringprivatelateinitvardataphototypedarrayprivatelateinitvardatalocationarraystringp
rivatelateinitvardatarepositoryarraystringprivatelateinitvardatafollowersarraystringprivatelateinitvardatafollowingarrayst
ringprivatevarusersarraylistofuseroverridefunoncreatesavedinstancestatebundle?s
uperoncreatesavedinstancestatesetcontentviewrlayoutactivitymainvalactionbarsupportactionbaractionbar                    !!
titlelistusergithubvallistviewlistviewfindviewbyidridlvlistadapteruseradapterthislistviewadapteradapterprepareadditemlistv
iewonitemclicklisteneradapterviewonitemclicklistenerposition                                                            -
valselecteduseruserspositionvalmoveintentwithparcelableintentthis@
mainactivitydetailuserclassjavamoveintentwithparcelableputextradetailuserextrauserselecteduserstartactivitymoveintentwithp
arcelableprivatefunpreparedatausernameresourcesgetstringarrayrarrayusernamedatanameresourcesgetstringarrayrarraynamedataco
mpanyresourcesgetstringarrayrarraycompanydataphotoresourcesobtaintypedarrayrarrayavatardatalocationresourcesgetstringarray
rarraylocationdatarepositoryresourcesgetstringarrayrarrayrepositorydatafollowersresourcesgetstringarrayrarrayfollowersdata
followingresourcesgetstringarrayrarrayfollowingprivatefunadditemforpositionindatanameindicesvaluseruserdatausernamepositio
ndatanamepositiondatacompanypositiondataphotogetresourceidposition                                  -                  1
datalocationpositiondatarepositorypositiondatafollowerspositiondatafollowingpositionusersadduseradapterusersusers
```

Source code 2

```
packagecommasuwesgithubuserfirstimportandroidcontentrestypedarrayimportandroidxappcompatappappcompatactivityimportandroido
sbundleimportandroidxrecyclerviewwidgetlinearlayoutmanagerimportandroidxrecyclerviewwidgetrecyclerviewimportkotlinxandroid
syntheticmainactivitymain                                                                                               *
classmainactivityappcompatactivityprivatelateinitvardatanamearraystringprivatelateinitvardatausernamearraystringprivatelat
einitvardatacompanyarraystringprivatelateinitvardatarepositoryarraystringprivatelateinitvardatafollowersarraystringprivate
lateinitvardatafollowingarraystringprivatelateinitvardataavatartypedarrayprivatelateinitvardatalocationarraystringprivatev
arusersarraylistofuserprivatelateinitvaradapteruseradapteroverridefunoncreatesavedinstancestatebundle?s
uperoncreatesavedinstancestatesetcontentviewrlayoutactivitymainvalactionbarsupportactionbaractionbar                    !!
settitleuserlistvallistviewrecyclerviewfindviewbyidridlistviewuseradapteruseradapterlistviewadapteradapterprepareadddatash
owrecyclerviewprivatefunpreparedatausernameresourcesgetstringarrayrarrayusernamedatanameresourcesgetstringarrayrarraynamed
atalocationresourcesgetstringarrayrarraylocationdatarepositoryresourcesgetstringarrayrarrayrepositorydatacompanyresourcesg
etstringarrayrarraycompanydatafollowersresourcesgetstringarrayrarrayfollowersdatafollowingresourcesgetstringarrayrarrayfol
lowingdataavatarresourcesobtaintypedarrayrarrayavatarprivatefunshowrecyclerviewvallayoutmanagerlinearlayoutmanagerthislist
viewuser?s                          etlayoutmanagerlayoutmanagerlistviewuser?s                          ethasfixedsizetrue
//thismethodareneedtomakesameorderlikethedataprivatefunadddataforpositionindatanameindicesvaluseruserdatausernamepositiond
```

```
atanamepositiondatalocationpositiondatarepositorypositiondatacompanypositiondatafollowerspositiondatafollowingpositiondata
avatargetresourceidposition-1usersadduseradapterlistuserusers
```

2. Establishment of k-gram values

The second step is to form k-grams as test sentences, the amount of n-gram grouping data starts from 2 to 10, here are the results of some test results using the value of k-gram = 6. The following is the source code used to cut the k number of characters displayed on Figure 3.2

```
text = str(text)
k_grams = []
for start in range(len(text) - self.k + 1):
    # only perform the work necessary if the current character is not a whitespace
    if text[start] != " ":
        end = self.k
        k_gram = text[start:end]
        # k_gram = self.clean_text(k_gram)
        # after cleaning up our k-gram, we can end up with a much smaller string that is below the denoted k-gram length
        # if this is the case, then we will continually add the missing length to the string we are slicing
        while(len(k_gram) < self.k):
            end = end + (self.k - len(k_gram)) # we will need to add as many letters as are missing from the k-gram
            k_gram = text[start: end]
            # k_gram = self.clean_text(k_gram)
        hashed_k_gram=self.custom_hash(k_gram)
        k_grams.append(hashed_k_gram)
        # print("K-gram:", k_gram)
        # print("Hash:", hashed_k_gram)
return k_grams
```

Figure 3.2 Source code for estabilishment the K-Gram

Source code 1

```
['packag', 'ackage', 'ckagec', 'kageco', 'agecom', 'gecoms', 'ecomsu', 'comsub', 'omsubm', 'msubmi', 'submis', 'ubmiss',
'bmissi', 'missio', 'ission', 'ssiong', 'siongi', 'iongit', 'ongith', 'ngithu', 'github', 'ithubu', 'thubus', 'hubuse',
'ubuser', 'busera', 'userap', 'serapp', 'erappl', 'rappli', 'applic', 'pplica', 'plicat', 'licati', 'icatio', 'cation',
'ationi', 'tionim', 'ionimp', 'onimpo', 'nimpor', 'import', 'mporta', 'portan', 'ortand', 'rtandr', 'tandro', 'androi',
'ndroid', 'droidc', 'roidco', 'oidcon', 'idcont', 'dconte', 'conten', 'ontent', 'ntenti', 'tentin', 'entint', 'ntinte',
'tinten', 'intent', 'ntenti', 'tentim', 'entimp', 'ntimpo', 'timpor', 'import', 'mporta', 'portan', 'ortand', 'rtandr',
'tandro', 'androi', 'ndroid', 'droidc', 'roidco', 'oidcon', 'idcont', 'dconte', 'conten', 'ontent', 'ntentr', 'tentre',
'entres', 'ntrest', 'tresty', 'restyp', 'estype', 'styped', 'typeda', 'ypedar', 'pedarr', 'edarra', 'darray', 'arrayi',
'rrayim', 'rayimp', 'ayimpo', 'yimpor', 'import', 'mporta', 'portan', 'ortand', 'rtandr', 'tandro', 'androi', 'ndroid',
'droido', 'roidos', 'oidosb', 'idosbu', 'dosbun', 'osbund', 'sbundl', 'bundle', 'undlei', 'ndleim', 'dleimp', 'leimpo',
'eimpor', 'import', 'mporta', 'portan', 'ortand', 'rtandr', 'tandro', 'androi', 'ndroid', 'droidw', 'roidwi', 'oidwid',
'idwidg', 'dwidge', 'widget', 'idgeta', 'dgetad', 'getada', 'etadap', 'tadapt', 'adapte', 'dapter', 'apterv', 'ptervi',
'tervie', 'erview', 'rviewi', 'viewim', 'iewimp', 'ewimpo', 'wimpor', 'import', 'mporta', 'portan', 'ortand', 'rtandr',
'tandro', 'androi', 'ndroid', 'droidw', 'roidwi', 'oidwid', 'idwidg', 'dwidge', 'widget', 'idgetl', 'dgetli', 'getlis',
'etlist', 'tlistv', 'listvi', 'istvie', 'stview', 'tviewi', 'viewim', 'iewimp', 'ewimpo', 'wimpor', 'import', 'mporta',
'portan', 'ortand', 'rtandr', 'tandro', 'androi', 'ndroid', 'droidx', 'roidxa', 'oidxap', 'idxapp', 'dxappc', 'xappco',
'appcom', 'ppcomp', 'pcompa', 'compat', 'ompata', 'mpatap', 'patapp', 'atappa', 'tappap', 'appapp', 'ppappc', 'pappco',
'appcom', 'ppcomp', 'pcompa', 'compat', 'ompata', 'mpatac', 'patact', 'atacti', 'tactiv', 'activi', 'ctivit', 'tivity',
'ivityc', 'vitycl', 'itycla', 'tyclas', 'yclass', 'classm', 'lassma', 'assmai', 'ssmain', 'smaina', 'mainac', 'ainact',
'inacti', 'nactiv', 'activi', 'ctivit', 'tivity', 'ivitya', 'vityap', 'ityapp', 'tyappc', 'yappco', 'appcom', 'ppcomp',
'pcompa', 'compat', 'ompata', 'mpatac', 'patact', 'atacti', 'tactiv', 'activi', 'ctivit', 'tivity', 'ivityp', 'vitypr',
'itypri', 'typriv', 'ypriva', 'privat', 'rivate', 'ivatel', 'vatela', 'atelat', 'telate', 'elatei', 'latein', 'ateini',
'teinit', 'einitv', 'initva', 'nitvar', 'itvara', 'tvarad', 'varada', 'aradap', 'radapt', 'adapte', 'dapter', 'apteru',
'pterus', 'teruse', 'eruser', 'rusera', 'userad', 'serada', 'eradap', 'radapt', 'adapte', 'dapter', 'apterp', 'pterpr',
'terpri', 'erpriv', 'rpriva', 'privat', 'rivate', 'ivatel', 'vatela', 'atelat', 'telate', 'elatei', 'latein', 'ateini',
'teinit', 'einitv', 'initva', 'nitvar', 'itvard', 'tvarda', 'vardat', 'ardata', 'rdatau', 'dataus', 'atause', 'tauser',
'ausern', 'userna', 'sernam', 'ername', 'rnamea', 'namear', 'amearr', 'mearra', 'earray', 'arrays', 'rrayst', 'raystr',
'aystri', 'ystrin', 'string', 'tringp', 'ringpr', 'ingpri', 'ngpriv', 'gpriva', 'privat', 'rivate', 'ivatel', 'vatela',
'atelat', 'telate', 'elatei', 'latein', 'ateini', 'teinit', 'einitv', 'initva', 'nitvar', 'itvard', 'tvarda', 'vardat',
'ardata', 'rdatan', 'datana', 'atanam', 'taname', 'anamea', 'namear', 'amearr', 'mearra', 'earray', 'arrays', 'rrayst',
'raystr', 'aystri', 'ystrin', 'string', 'tringp', 'ringpr', 'ingpri', 'ngpriv', 'gpriva', 'privat', 'rivate', 'ivatel',
'vatela', 'atelat', 'telate', 'elatei', 'latein', 'ateini', 'teinit', 'einitv', 'initva', 'nitvar', 'itvard', 'tvarda',
'vardat', 'ardata', 'rdatac', 'dataco', 'atacom', 'tacomp', 'acompa', 'compan', 'ompany', 'mpanya', 'panyar', 'anyarr',
'nyarra', 'yarray', 'arrays', 'rrayst', 'raystr', 'aystri', 'ystrin', 'string', 'tringp', 'ringpr', 'ingpri', 'ngpriv',
'gpriva', 'privat', 'rivate', 'ivatel', 'vatela', 'atelat', 'telate', 'elatei', 'latein', 'ateini', 'teinit', 'einitv',
'initva', 'nitvar', 'itvard', 'tvarda', 'vardat', 'ardata', 'rdatap', 'dataph', 'atapho', 'taphot', 'aphoto', 'photot',
'hototy', 'ototyp', 'totype', 'otyped', 'typeda', 'ypedar', 'pedarr', 'edarra', 'darray', 'arrayp', 'rraypr', 'raypri',
'aypriv', 'ypriva', 'privat', 'rivate', 'ivatel', 'vatela', 'atelat', 'telate', 'elatei', 'latein', 'ateini', 'teinit',
'einitv', 'initva', 'nitvar', 'itvard', 'tvarda', 'vardat', 'ardata', 'rdatal', 'datalo', 'ataloc', 'taloca', 'alocat',
'locati', 'ocatio', 'cation', 'ationa', 'tionar', 'ionarr', 'onarra', 'narray', 'arrays', 'rrayst', 'raystr', 'aystri',
'ystrin', 'string', 'tringp', 'ringpr', 'ingpri', 'ngpriv', 'gpriva', 'privat', 'rivate', 'ivatel', 'vatela', 'atelat',
'telate', 'elatei', 'latein', 'ateini', 'teinit', 'einitv', 'initva', 'nitvar', 'itvard', 'tvarda', 'vardat', 'ardata',
'rdatar', 'datare', 'atarep', 'tarepo', 'arepos', 'reposi', 'eposit', 'posito', 'ositor', 'sitory', 'itorya', 'toryar',
'oryarr', 'ryarra', 'yarray', 'arrays', 'rrayst', 'raystr', 'aystri', 'ystrin', 'string', 'tringp', 'ringpr', 'ingpri',
'ngpriv', 'gpriva', 'privat', 'rivate', 'ivatel', 'vatela', 'atelat', 'telate', 'elatei', 'latein', 'ateini', 'teinit',
'einitv', 'initva', 'nitvar', 'itvard', 'tvarda', 'vardat', 'ardata', 'rdataf', 'datafo', 'atafol', 'tafoll', 'afollo',
'follow', 'ollowe', 'llower', 'lowers', 'owersa', 'wersar', 'ersarr', 'rsarra', 'sarray', 'arrays', 'rrayst', 'raystr',
'aystri', 'ystrin', 'string', 'tringp', 'ringpr', 'ingpri', 'ngpriv', 'gpriva', 'privat', 'rivate', 'ivatel', 'vatela',
'atelat', 'telate', 'elatei', 'latein', 'ateini', 'teinit', 'einitv', 'initva', 'nitvar', 'itvard', 'tvarda', 'vardat',
'ardata', 'rdataf', 'datafo', 'atafol', 'tafoll', 'afollo', 'follow', 'ollowi', 'llowin', 'lowing', 'owinga', 'wingar',
'ingarr', 'ngarra', 'garray', 'arrays', 'rrayst', 'raystr', 'aystri', 'ystrin', 'string', 'tringp', 'ringpr', 'ingpri',
'ngpriv', 'gpriva', 'privat', 'rivate', 'ivatev', 'vateva', 'atevar', 'tevaru', 'evarus', 'varuse', 'aruser', 'rusers',
'usersa', 'sersar', 'ersarr', 'rsarra', 'sarray', 'arrayl', 'rrayli', 'raylis', 'aylist', 'ylisto', 'listof', 'istofu',
'stofus', 'tofuse', 'ofuser', 'fusero', 'userov', 'serove', 'erover', 'roverr', 'overri', 'verrid', 'erride', 'rridef',
'ridefu', 'idefun', 'defuno', 'efunon', 'funonc', 'unoncr', 'noncre', 'oncrea', 'ncreat', 'create', 'reates', 'eatesa',
'atesav', 'tesave', 'esaved', 'savedi', 'avedin', 'vedins', 'edinst', 'dinsta', 'instan', 'nstanc', 'stance', 'tances',
'ancest', 'ncesta', 'cestat', 'estate', 'stateb', 'tatebu', 'atebun', 'tebund', 'ebundl', 'bundle', 'undle?', 'ndle?s',
'dle?s ', 'le?s u', 'e?s up', '?s upe', 's uper', ' upero', 'peronc', 'eroncr', 'roncre', 'oncrea', 'ncreat', 'create',
'reates', 'eatesa', 'atesav', 'tesave', 'esaved', 'savedi', 'avedin', 'vedins', 'edinst', 'dinsta', 'instan', 'nstanc',
```

```
'stance', 'tances', 'ancest', 'ncesta', 'cestat', 'estate', 'states', 'tatese', 'ateset', 'tesetc', 'esetco', 'setcon',
'etcont', 'tconte', 'conten', 'ontent', 'ntentv', 'tentvi', 'entvie', 'ntview', 'tviewr', 'viewrl', 'iewrla', 'ewrlay',
'wrlayo', 'rlayou', 'layout', 'ayouta', 'youtac', 'outact', 'utacti', 'tactiv', 'activi', 'ctivit', 'tivity', 'ivitym',
'vityma', 'itymai', 'tymain', 'ymainv', 'mainva', 'ainval', 'invala', 'nvalac', 'valact', 'alacti', 'lactio', 'action',
'ctionb', 'tionba', 'ionbar', 'onbars', 'nbarsu', 'barsup', 'arsupp', 'rsuppo', 'suppor', 'upport', 'pporta', 'portac',
'ortact', 'rtacti', 'tactio', 'action', 'ctionb', 'tionba', 'ionbar', 'onbara', 'nbarac', 'baract', 'aracti', 'ractio',
'action', 'ctionb', 'tionba', 'ionbar', 'onbar ', 'nbar !', 'bar !!', 'ar !! ', 'r !! t', ' !! ti', '! titl', ' title',
'itleli', 'tlelis', 'lelist', 'elistu', 'listus', 'istuse', 'stuser', 'tuserg', 'usergi', 'sergit', 'ergith', 'rgithu',
'github', 'ithubv', 'thubva', 'hubval', 'ubvall', 'bvalli', 'vallis', 'allist', 'llistv', 'listvi', 'istvie', 'stview',
'tviewl', 'viewli', 'iewlis', 'ewlist', 'wlistv', 'listvi', 'istvie', 'stview', 'tviewf', 'viewfi', 'iewfin', 'ewfind',
'wfindv', 'findvi', 'indvie', 'ndview', 'dviewb', 'viewby', 'iewbyi', 'ewbyid', 'wbyidr', 'byidri', 'yidrid', 'idridl',
'dridlv', 'ridlvl', 'idlvli', 'dlvlis', 'lvlist', 'vlista', 'listad', 'istada', 'stadap', 'tadapt', 'adapte', 'dapter',
'apteru', 'pterus', 'teruse', 'eruser', 'rusera', 'userad', 'serada', 'eradap', 'radapt', 'adapte', 'dapter', 'aptert',
'pterth', 'terthi', 'erthis', 'rthisl', 'thisli', 'hislis', 'islist', 'slistv', 'listvi', 'istvie', 'stview', 'tviewa',
'viewad', 'iewada', 'ewadap', 'wadapt', 'adapte', 'dapter', 'aptera', 'pterad', 'terada', 'eradap', 'radapt', 'adapte',
'dapter', 'apterp', 'pterpr', 'terpre', 'erprep', 'rprepa', 'prepar', 'repare', 'eparea', 'paread', 'areadd', 'readdi',
'eaddit', 'addite', 'dditem', 'diteml', 'itemli', 'temlis', 'emlist', 'mlistv', 'listvi', 'istvie', 'stview', 'tviewo',
'viewon', 'iewoni', 'ewonit', 'wonite', 'onitem', 'nitemc', 'itemcl', 'temcli', 'emclic', 'mclick', 'clickl', 'lickli',
'icklis', 'cklist', 'kliste', 'listen', 'istene', 'stener', 'tenera', 'enerad', 'nerada', 'eradap', 'radapt', 'adapte',
'dapter', 'apterv', 'ptervi', 'tervie', 'erview', 'rviewo', 'viewon', 'iewoni', 'ewonit', 'wonite', 'onitem', 'nitemc',
'itemcl', 'temcli', 'emclic', 'mclick', 'clickl', 'lickli', 'icklis', 'cklist', 'kliste', 'listen', 'istene', 'stener',
'tenerp', 'enerpo', 'nerpos', 'erposi', 'rposit', 'positi', 'ositio', 'sition', 'ition ', 'tion -', 'ion - ', 'on - v',
'n - va', ' - val', ' valse', 'alsele', 'lselec', 'select', 'electe', 'lected', 'ectedu', 'ctedus', 'teduse', 'eduser',
'duseru', 'userus', 'seruse', 'eruser', 'ruseru', 'userus', 'seruse', 'eruser', 'rusers', 'usersp', 'serspo', 'erspos',
'rsposi', 'sposit', 'positi', 'ositio', 'sition', 'itionv', 'tionva', 'ionval', 'onvalm', 'nvalmo', 'valmov', 'almove',
'lmovei', 'movein', 'oveint', 'veinte', 'einten', 'intent', 'ntentw', 'tentwi', 'entwit', 'ntwith', 'twithp', 'withpa',
'ithpar', 'thparc', 'hparce', 'parcel', 'arcela', 'rcelab', 'celabl', 'elable', 'lablei', 'ablein', 'bleint', 'leinte',
'einten', 'intent', 'ntentt', 'tentth', 'entthi', 'ntthis', 'tthis@', 'this@ ', 'his@ m', 'is@ ma', 's@ mai', '@ main',
' maina', 'ainact', 'inacti', 'nactiv', 'activi', 'ctivit', 'tivity', 'ivityd', 'vityde', 'itydet', 'tydeta', 'ydetai',
'detail', 'etailu', 'tailus', 'ailuse', 'iluser', 'luserc', 'usercl', 'sercla', 'erclas', 'rclass', 'classj', 'lassja',
'assjav', 'ssjava', 'sjavam', 'javamo', 'avamov', 'vamove', 'amovei', 'movein', 'oveint', 'veinte', 'einten', 'intent',
'ntentw', 'tentwi', 'entwit', 'ntwith', 'twithp', 'withpa', 'ithpar', 'thparc', 'hparce', 'parcel', 'arcela', 'rcelab',
'celabl', 'elable', 'lablep', 'ablepu', 'bleput', 'lepute', 'eputex', 'putext', 'utextr', 'textra', 'extrad', 'xtrade',
'tradet', 'radeta', 'adetai', 'detail', 'etailu', 'tailus', 'ailuse', 'iluser', 'lusere', 'userex', 'serext', 'erextr',
'rextra', 'extrau', 'xtraus', 'trause', 'rauser', 'ausers', 'userse', 'sersel', 'ersele', 'rselec', 'select', 'electe',
'lected', 'ectedu', 'ctedus', 'teduse', 'eduser', 'dusers', 'userst', 'sersta', 'erstar', 'rstart', 'starta', 'tartac',
'artact', 'rtacti', 'tactiv', 'activi', 'ctivit', 'tivity', 'ivitym', 'vitymo', 'itymov', 'tymove', 'ymovei', 'movein',
'oveint', 'veinte', 'einten', 'intent', 'ntentw', 'tentwi', 'entwit', 'ntwith', 'twithp', 'withpa', 'ithpar', 'thparc',
'hparce', 'parcel', 'arcela', 'rcelab', 'celabl', 'elable', 'lablep', 'ablepr', 'blepri', 'lepriv', 'epriva', 'privat',
'rivate', 'ivatef', 'vatefu', 'atefun', 'tefunp', 'efunpr', 'funpre', 'unprep', 'nprepa', 'prepar', 'repare', 'epared',
'pareda', 'aredat', 'redata', 'edatau', 'dataus', 'atause', 'tauser', 'ausern', 'userna', 'sernam', 'ername', 'rnamer',
'namere', 'ameres', 'mereso', 'eresou', 'resour', 'esourc', 'source', 'ources', 'urcesg', 'rcesge', 'cesget', 'esgets',
'sgetst', 'getstr', 'etstri', 'tstrin', 'string', 'tringa', 'ringar', 'ingarr', 'ngarra', 'garray', 'arrayr', 'rrayra',
'rayrar', 'ayrarr', 'yrarra', 'rarray', 'arrayu', 'rrayus', 'rayuse', 'ayuser', 'yusern', 'userna', 'sernam', 'ername',
'rnamed', 'nameda', 'amedat', 'medata', 'edatan', 'datana', 'atanam', 'taname', 'anamer', 'namere', 'ameres', 'mereso',
'eresou', 'resour', 'esourc', 'source', 'ources', 'urcesg', 'rcesge', 'cesget', 'esgets', 'sgetst', 'getstr', 'etstri',
'tstrin', 'string', 'tringa', 'ringar', 'ingarr', 'ngarra', 'garray', 'arrayr', 'rrayra', 'rayrar', 'ayrarr', 'yrarra',
'rarray', 'arrayn', 'rrayna', 'raynam', 'ayname', 'ynamed', 'nameda', 'amedat', 'medata', 'edatac', 'dataco', 'atacom',
'tacomp', 'acompa', 'compan', 'ompany', 'mpanyr', 'panyre', 'anyres', 'nyreso', 'yresou', 'resour', 'esourc', 'source',
'ources', 'urcesg', 'rcesge', 'cesget', 'esgets', 'sgetst', 'getstr', 'etstri', 'tstrin', 'string', 'tringa', 'ringar',
'ingarr', 'ngarra', 'garray', 'arrayr', 'rrayra', 'rayrar', 'ayrarr', 'yrarra', 'rarray', 'arrayc', 'rrayco', 'raycom',
'aycomp', 'ycompa', 'compan', 'ompany', 'mpanyd', 'panyda', 'anydat', 'nydata', 'ydatap', 'dataph', 'atapho', 'taphot',
'aphoto', 'photor', 'hotore', 'otores', 'toreso', 'oresou', 'resour', 'esourc', 'source', 'ources', 'urceso', 'rcesob',
'cesobt', 'esobta', 'sobtai', 'obtain', 'btaint', 'tainty', 'aintyp', 'intype', 'ntyped', 'typeda', 'ypedar', 'pedarr',
'edarra', 'darray', 'arrayr', 'rrayra', 'rayrar', 'ayrarr', 'yrarra', 'rarray', 'arraya', 'rrayav', 'rayava', 'ayavat',
'yavata', 'avatar', 'vatard', 'atarda', 'tardat', 'ardata', 'rdatal', 'datalo', 'ataloc', 'taloca', 'alocat', 'locati',
'ocatio', 'cation', 'ationr', 'tionre', 'ionres', 'onreso', 'nresou', 'resour', 'esourc', 'source', 'ources', 'urcesg',
'rcesge', 'cesget', 'esgets', 'sgetst', 'getstr', 'etstri', 'tstrin', 'string', 'tringa', 'ringar', 'ingarr', 'ngarra',
'garray', 'arrayr', 'rrayra', 'rayrar', 'ayrarr', 'yrarra', 'rarray', 'arrayl', 'rraylo', 'rayloc', 'ayloca', 'ylocat',
'locati', 'ocatio', 'cation', 'ationd', 'tionda', 'iondat', 'ondata', 'ndatar', 'datare', 'atarep', 'tarepo', 'arepos',
'reposi', 'eposit', 'posito', 'ositor', 'sitory', 'itoryr', 'toryre', 'oryres', 'ryreso', 'yresou', 'resour', 'esourc',
'source', 'ources', 'urcesg', 'rcesge', 'cesget', 'esgets', 'sgetst', 'getstr', 'etstri', 'tstrin', 'string', 'tringa',
'ringar', 'ingarr', 'ngarra', 'garray', 'arrayr', 'rrayra', 'rayrar', 'ayrarr', 'yrarra', 'rarray', 'arrayr', 'rrayre',
'rayrep', 'ayrepo', 'yrepos', 'reposi', 'eposit', 'posito', 'ositor', 'sitory', 'itoryd', 'toryda', 'orydat', 'rydata',
'ydataf', 'datafo', 'atafol', 'tafoll', 'afollo', 'follow', 'ollowe', 'llower', 'lowers', 'owersr', 'wersre', 'ersres',
'rsreso', 'sresou', 'resour', 'esourc', 'source', 'ources', 'urcesg', 'rcesge', 'cesget', 'esgets', 'sgetst', 'getstr',
'etstri', 'tstrin', 'string', 'tringa', 'ringar', 'ingarr', 'ngarra', 'garray', 'arrayr', 'rrayra', 'rayrar', 'ayrarr',
'yrarra', 'rarray', 'arrayf', 'rrayfo', 'rayfol', 'ayfoll', 'yfollo', 'follow', 'ollowe', 'llower', 'lowers', 'owersd',
'wersda', 'ersdat', 'rsdata', 'sdataf', 'datafo', 'atafol', 'tafoll', 'afollo', 'follow', 'ollowi', 'llowin', 'lowing',
'owingr', 'wingre', 'ingres', 'ngreso', 'gresou', 'resour', 'esourc', 'source', 'ources', 'urcesg', 'rcesge', 'cesget',
'esgets', 'sgetst', 'getstr', 'etstri', 'tstrin', 'string', 'tringa', 'ringar', 'ingarr', 'ngarra', 'garray', 'arrayr',
'rrayra', 'rayrar', 'ayrarr', 'yrarra', 'rarray', 'arrayf', 'rrayfo', 'rayfol', 'ayfoll', 'yfollo', 'follow', 'ollowi',
'llowin', 'lowing', 'owingp', 'wingpr', 'ingpri', 'ngpriv', 'gpriva', 'privat', 'rivate', 'ivatef', 'vatefu', 'atefun',
'tefuna', 'efunad', 'funadd', 'unaddi', 'naddit', 'addite', 'dditem', 'ditemf', 'itemfo', 'temfor', 'emforp', 'mforpo',
'forpos', 'orposi', 'rposit', 'positi', 'ositio', 'sition', 'itioni', 'tionin', 'ionind', 'oninda', 'nindat', 'indata',
'ndatan', 'datana', 'atanam', 'taname', 'anamei', 'namein', 'ameind', 'meindi', 'eindic', 'indice', 'ndices', 'dicesv',
'icesva', 'cesval', 'esvalu', 'svalus', 'valuse', 'aluser', 'luseru', 'userus', 'seruse', 'eruser', 'ruserd', 'userda',
'serdat', 'erdata', 'rdatau', 'dataus', 'atause', 'tauser', 'ausern', 'userna', 'sernam', 'ername', 'rnamep', 'namepo',
'amepos', 'meposi', 'eposit', 'positi', 'ositio', 'sition', 'itiond', 'tionda', 'iondat', 'ondata', 'ndatan', 'datana',
'atanam', 'taname', 'anamep', 'namepo', 'amepos', 'meposi', 'eposit', 'positi', 'ositio', 'sition', 'itiond', 'tionda',
'iondat', 'ondata', 'ndatac', 'dataco', 'atacom', 'tacomp', 'acompa', 'compan', 'ompany', 'mpanyp', 'panypo', 'anypos',
'nyposi', 'yposit', 'positi', 'ositio', 'sition', 'itiond', 'tionda', 'iondat', 'ondata', 'ndatap', 'dataph', 'atapho',
'taphot', 'aphoto', 'photog', 'hotoge', 'otoget', 'togetr', 'ogetre', 'getres', 'etreso', 'tresou', 'resour', 'esourc',
'source', 'ourcei', 'urceid', 'rceidp', 'ceidpo', 'eidpos', 'idposi', 'dposit', 'positi', 'ositio', 'sition', 'ition ',
'tion -', 'ion - ', 'on - 1', 'n - 1 ', ' - 1 d', ' 1 dat', ' datal', 'ataloc', 'taloca', 'alocat', 'locati', 'ocatio',
'cation', 'ationp', 'tionpo', 'ionpos', 'onposi', 'nposit', 'positi', 'ositio', 'sition', 'itiond', 'tionda', 'iondat',
'ondata', 'ndatar', 'datare', 'atarep', 'tarepo', 'arepos', 'reposi', 'eposit', 'posito', 'ositor', 'sitory', 'itoryp',
'torypo', 'orypos', 'ryposi', 'yposit', 'positi', 'ositio', 'sition', 'itiond', 'tionda', 'iondat', 'ondata', 'ndataf',
'datafo', 'atafol', 'tafoll', 'afollo', 'follow', 'ollowe', 'llower', 'lowers', 'owersp', 'werspo', 'erspos', 'rsposi',
'sposit', 'positi', 'ositio', 'sition', 'itiond', 'tionda', 'iondat', 'ondata', 'ndataf', 'datafo', 'atafol', 'tafoll',
'afollo', 'follow', 'ollowi', 'llowin', 'lowing', 'owingp', 'wingpo', 'ingpos', 'ngposi', 'gposit', 'positi', 'ositio',
'sition', 'itionu', 'tionus', 'ionuse', 'onuser', 'nusers', 'usersa', 'sersad', 'ersadd', 'rsaddu', 'saddus', 'adduse',
'dduser', 'dusera', 'userad', 'serada', 'eradap', 'radapt', 'adapte', 'dapter', 'apteru', 'pterus', 'teruse', 'eruser',
'rusers', 'usersu', 'sersus', 'ersuse', 'rsuser', 'susers', 'users ', 'sers \n']
```

Source code 2

```
['packag', 'ackage', 'ckagec', 'kageco', 'agecom', 'gecomm', 'ecomma', 'commas', 'ommasu', 'mmasuw', 'masuwe', 'asuwes',
'suwesg', 'uwesgi', 'wesgit', 'esgith', 'sgithu', 'github', 'ithubu', 'thubus', 'hubuse', 'ubuser', 'buserf', 'userfi',
'serfir', 'erfirs', 'rfirst', 'firsti', 'irstim', 'rstimp', 'stimpo', 'timpor', 'impor', 'mporta', 'portan', 'ortand',
'rtandr', 'tandro', 'androi', 'ndroid', 'droidc', 'roidco', 'oidcon', 'idcont', 'dconte', 'conten', 'ontent', 'ntentr',
'tentre', 'entres', 'ntrest', 'tresty', 'restyp', 'estype', 'styped', 'typeda', 'ypedar', 'pedarr', 'edarra', 'darray',
'arrayi', 'rrayim', 'rayimp', 'ayimpo', 'yimpor', 'import', 'mporta', 'portan', 'ortand', 'rtandr', 'tandro', 'androi',
'ndroid', 'droidx', 'roidxa', 'oidxap', 'idxapp', 'dxappc', 'xappco', 'appcom', 'ppcomp', 'pcompa', 'compat', 'ompata',
'mpatap', 'patapp', 'atappa', 'tappap', 'appapp', 'ppappc', 'pappco', 'appcom', 'ppcomp', 'pcompa', 'compat', 'ompata',
'mpatac', 'patact', 'atacti', 'tactiv', 'activi', 'ctivit', 'tivity', 'ivityi', 'vityim', 'ityimp', 'tyimpo', 'yimpor',
'import', 'mporta', 'portan', 'ortand', 'rtandr', 'tandro', 'androi', 'ndroid', 'droido', 'roidos', 'oidosb', 'idosbu',
'dosbun', 'osbund', 'sbundl', 'bundle', 'undlei', 'ndleim', 'dleimp', 'leimpo', 'eimpor', 'import', 'mporta', 'portan',
'ortand', 'rtandr', 'tandro', 'androi', 'ndroid', 'droidx', 'roidxr', 'oidxre', 'idxrec', 'dxrecy', 'xrecyc', 'recycl',
'ecycle', 'cycler', 'yclerv', 'clervi', 'lervie', 'erview', 'rvieww', 'viewwi', 'iewwid', 'ewwidg', 'wwidge', 'widget',
'idgetl', 'dgetli', 'getlin', 'etline', 'tlinea', 'linear', 'inearl', 'nearla', 'earlay', 'arlayo', 'rlayou', 'layout',
'ayoutm', 'youtma', 'outman', 'utmana', 'tmanag', 'manage', 'anager', 'nageri', 'agerim', 'gerimp', 'erimpo', 'rimpor',
'import', 'mporta', 'portan', 'ortand', 'rtandr', 'tandro', 'androi', 'ndroid', 'droidx', 'roidxr', 'oidxre', 'idxrec',
'dxrecy', 'xrecyc', 'recycl', 'ecycle', 'cycler', 'yclerv', 'clervi', 'lervie', 'erview', 'rvieww', 'viewwi', 'iewwid',
'ewwidg', 'wwidge', 'widget', 'idgetr', 'dgetre', 'getrec', 'etrecy', 'trecyc', 'recycl', 'ecycle', 'cycler', 'yclerv',
'clervi', 'lervie', 'erview', 'rviewi', 'viewim', 'iewimp', 'ewimpo', 'wimpor', 'import', 'mportk', 'portko', 'ortkot',
'rtkotl', 'tkotli', 'kotlin', 'otlinx', 'tlinxa', 'linxan', 'inxand', 'nxandr', 'xandro', 'androi', 'ndroid', 'droids',
'roidsy', 'oidsyn', 'idsynt', 'dsynth', 'synthe', 'ynthet', 'ntheti', 'thetic', 'heticm', 'eticma', 'ticmai', 'icmain',
'cmaina', 'mainac', 'ainact', 'inacti', 'nactiv', 'activi', 'ctivit', 'tivity', 'ivitym', 'vityma', 'itymai', 'tymain',
'ymain ', 'main *', 'ain * ', 'in * c', 'n * cl', ' * cla', ' class', 'lassma', 'assmai', 'ssmain', 'smaina', 'mainac',
'ainact', 'inacti', 'nactiv', 'activi', 'ctivit', 'tivity', 'ivitya', 'vityap', 'ityapp', 'tyappc', 'yappco', 'appcom',
'ppcomp', 'pcompa', 'compat', 'ompata', 'mpatac', 'patact', 'atacti', 'tactiv', 'activi', 'ctivit', 'tivity', 'ivityp',
'vitypr', 'itypri', 'typriv', 'ypriva', 'privat', 'rivate', 'ivatel', 'vatela', 'atelat', 'telate', 'elatei', 'latein',
'ateini', 'teinit', 'einitv', 'initva', 'nitvar', 'itvard', 'tvarda', 'vardat', 'ardata', 'rdatan', 'datana', 'atanam',
'taname', 'anamea', 'namear', 'amearr', 'mearra', 'earray', 'arrays', 'rrayst', 'raystr', 'aystri', 'ystrin', 'string',
'tringp', 'ringpr', 'ingpri', 'ngpriv', 'gpriva', 'privat', 'rivate', 'ivatel', 'vatela', 'atelat', 'telate', 'elatei',
'latein', 'ateini', 'teinit', 'einitv', 'initva', 'nitvar', 'itvard', 'tvarda', 'vardat', 'ardata', 'rdatau', 'dataus',
'atause', 'tauser', 'ausern', 'userna', 'sernam', 'ername', 'rnamea', 'namear', 'amearr', 'mearra', 'earray', 'arrays',
'rrayst', 'raystr', 'aystri', 'ystrin', 'string', 'tringp', 'ringpr', 'ingpri', 'ngpriv', 'gpriva', 'privat', 'rivate',
'ivatel', 'vatela', 'atelat', 'telate', 'elatei', 'latein', 'ateini', 'teinit', 'einitv', 'initva', 'nitvar', 'itvard',
'tvarda', 'vardat', 'ardata', 'rdatac', 'dataco', 'atacom', 'tacomp', 'acompa', 'compan', 'ompany', 'mpanya', 'panyar',
'anyarr', 'nyarra', 'yarray', 'arrays', 'rrayst', 'raystr', 'aystri', 'ystrin', 'string', 'tringp', 'ringpr', 'ingpri',
'ngpriv', 'gpriva', 'privat', 'rivate', 'ivatel', 'vatela', 'atelat', 'telate', 'elatei', 'latein', 'ateini', 'teinit',
'einitv', 'initva', 'nitvar', 'itvard', 'tvarda', 'vardat', 'ardata', 'rdatar', 'datare', 'atarep', 'tarepo', 'arepos',
'reposi', 'eposit', 'posito', 'ositor', 'sitory', 'itorya', 'toryar', 'oryarr', 'ryarra', 'yarray', 'arrays', 'rrayst',
'raystr', 'aystri', 'ystrin', 'string', 'tringp', 'ringpr', 'ingpri', 'ngpriv', 'gpriva', 'privat', 'rivate', 'ivatel',
'vatela', 'atelat', 'telate', 'elatei', 'latein', 'ateini', 'teinit', 'einitv', 'initva', 'nitvar', 'itvard', 'tvarda',
'vardat', 'ardata', 'rdataf', 'datafo', 'atafol', 'tafoll', 'afollo', 'follow', 'ollowe', 'llower', 'lowers', 'owersa',
'wersar', 'ersarr', 'rsarra', 'sarray', 'arrays', 'rrayst', 'raystr', 'aystri', 'ystrin', 'string', 'tringp', 'ringpr',
'ingpri', 'ngpriv', 'gpriva', 'privat', 'rivate', 'ivatel', 'vatela', 'atelat', 'telate', 'elatei', 'latein', 'ateini',
'teinit', 'einitv', 'initva', 'nitvar', 'itvard', 'tvarda', 'vardat', 'ardata', 'rdatal', 'datalo', 'ataloc', 'taloca', 'alocat', 'locati', 'ocatio', 'cation',
'ationa', 'tionar', 'ionarr', 'onarra', 'narray', 'arrays', 'rrayst', 'raystr', 'aystri', 'ystrin', 'string', 'tringp',
'ringpr', 'ingpri', 'ngpriv', 'gpriva', 'privat', 'rivate', 'ivatev', 'vateva', 'atevar', 'tevaru', 'evarus', 'varuse',
'aruser', 'rusers', 'usersa', 'sersar', 'ersarr', 'rsarra', 'sarray', 'arrayl', 'rrayli', 'raylis', 'aylist', 'ylisto',
'listof', 'istofu', 'stofus', 'tofuse', 'ofuser', 'fuserp', 'userpr', 'serpri', 'erpriv', 'rpriva', 'priva', 'rivate',
'ivatel', 'vatela', 'atelat', 'telate', 'elatei', 'latein', 'ateini', 'teinit', 'einitv', 'initva', 'nitvar', 'itvara',
'tvarad', 'varada', 'aradap', 'radapt', 'adapte', 'dapter', 'apteru', 'pterus', 'teruse', 'eruser', 'rusera', 'userad',
'serada', 'eradap', 'radapt', 'adapte', 'dapter', 'aptero', 'pterov', 'terove', 'erover', 'roverr', 'overri', 'verrid',
'erride', 'rridef', 'ridefu', 'idefun', 'defuno', 'efunon', 'funonc', 'unoncr', 'noncre', 'oncrea', 'ncreat', 'create',
'reates', 'eatesa', 'atesav', 'tesave', 'esaved', 'savedi', 'avedin', 'vedins', 'edinst', 'dinsta', 'instan', 'nstanc',
'stance', 'tances', 'ancest', 'ncesta', 'cestat', 'estate', 'stateb', 'tatebu', 'atebun', 'tebund', 'ebundl', 'bundle',
'undle?', 'ndle?s', 'dle?s ', 'le?s u', 'e?s up', '?s upe', 's uper', ' upero', 'peronc', 'eroncr', 'roncre', 'oncrea',
'ncreat', 'create', 'reates', 'eatesa', 'atesav', 'tesave', 'esaved', 'savedi', 'avedin', 'vedins', 'edinst', 'dinsta',
'instan', 'nstanc', 'stance', 'tances', 'ancest', 'ncesta', 'cestat', 'estate', 'states', 'tatese', 'ateset', 'tesetc',
'esetco', 'setcon', 'etcont', 'tconte', 'conten', 'ontent', 'ntentv', 'tentvi', 'entvie', 'ntview', 'tviewr', 'viewrl',
'iewrla', 'ewrlay', 'wrlayo', 'rlayou', 'layout', 'ayouta', 'youtac', 'outact', 'utacti', 'tactiv', 'activi', 'ctivit',
'tivity', 'ivitym', 'vityma', 'itymai', 'tymain', 'ymainv', 'mainva', 'ainval', 'invala', 'nvalac', 'valact', 'alacti',
'lactio', 'action', 'ctionb', 'tionba', 'ionbar', 'onbars', 'nbarsu', 'barsup', 'arsupp', 'rsuppo', 'suppor', 'upport',
'pporta', 'portac', 'ortact', 'rtacti', 'tactio', 'action', 'ctionb', 'tionba', 'ionbar', 'onbara', 'nbarac', 'baract',
'aracti', 'ractio', 'action', 'ctionb', 'tionba', 'ionbar', 'onbar ', 'nbar !', 'bar !!', 'ar !! ', 'r !! s', ' !! se', '!
sett', ' setti', 'ettitl', 'ttitle', 'titleu', 'itleus', 'tleuse', 'leuser', 'euserl', 'userli', 'serlis', 'erlist',
'rlistv', 'listva', 'istval', 'stvall', 'tvalli', 'vallis', 'allist', 'llistv', 'listvi', 'istvie', 'stview', 'tviewr',
'viewre', 'iewrec', 'ewrecy', 'wrecyc', 'recycl', 'ecycle', 'cycler', 'yclerv', 'clervi', 'lervie', 'erview', 'rviewf',
'viewfi', 'iewfin', 'ewfind', 'wfindv', 'findvi', 'indvie', 'ndview', 'dviewb', 'viewby', 'iewbyi', 'ewbyid', 'wbyidr',
'byidri', 'yidrid', 'idridl', 'dridli', 'ridlis', 'idlist', 'dlistv', 'listvi', 'istvie', 'stview', 'tviewu', 'viewus',
'iewuse', 'ewuser', 'wusera', 'userad', 'serada', 'eradap', 'radapt', 'adapte', 'dapter', 'apteru', 'pterus', 'teruse',
'eruser', 'rusera', 'userad', 'serada', 'eradap', 'radapt', 'adapte', 'dapter', 'apterl', 'pterli', 'terlis', 'erlist',
'rlistv', 'listvi', 'istvie', 'stview', 'tviewa', 'viewad', 'iewada', 'ewadap', 'wadapt', 'adapte', 'dapter', 'aptera',
'pterad', 'terada', 'eradap', 'radapt', 'adapte', 'dapter', 'apterp', 'pterpr', 'terpre', 'erprep', 'rprepa', 'prepar',
'repare', 'eparea', 'paread', 'areadd', 'readdd', 'eaddda', 'adddat', 'dddata', 'ddatas', 'datash', 'atasho', 'tashow',
'ashowr', 'showre', 'howrec', 'owrecy', 'wrecyc', 'recycl', 'ecycle', 'cycler', 'yclerv', 'clervi', 'lervie', 'erview',
'rviewp', 'viewpr', 'iewpri', 'ewpriv', 'wpriva', 'privat', 'rivate', 'ivatef', 'vatefu', 'atefun', 'tefunp', 'efunpr',
'funpre', 'unprep', 'nprepa', 'prepar', 'repare', 'epared', 'pareda', 'aredat', 'redata', 'edatau', 'dataus', 'atause',
'tauser', 'ausern', 'userna', 'sernam', 'ername', 'rnamer', 'namere', 'ameres', 'mereso', 'eresou', 'resour', 'esourc',
'source', 'ources', 'urcesg', 'rcesge', 'cesget', 'esgets', 'sgetst', 'getstr', 'etstri', 'tstrin', 'string', 'tringa',
'ringar', 'ingarr', 'ngarra', 'garray', 'arrayr', 'rrayra', 'rayrar', 'ayrarr', 'yrarra', 'rarray', 'arrayu', 'rrayus',
'rayuse', 'ayuser', 'yusern', 'userna', 'sernam', 'ername', 'rnamed', 'nameda', 'amedat', 'medata', 'edatan', 'datana',
'atanam', 'taname', 'anamer', 'namere', 'ameres', 'mereso', 'eresou', 'resour', 'esourc', 'source', 'ources', 'urcesg',
'rcesge', 'cesget', 'esgets', 'sgetst', 'getstr', 'etstri', 'tstrin', 'string', 'tringa', 'ringar', 'ingarr', 'ngarra',
'garray', 'arrayr', 'rrayra', 'rayrar', 'ayrarr', 'yrarra', 'rarray', 'arrayn', 'rrayna', 'raynam', 'ayname', 'ynamed',
'nameda', 'amedat', 'medata', 'edatal', 'datalo', 'ataloc', 'taloca', 'alocat', 'locati', 'ocatio', 'cation', 'ationr',
'tionre', 'ionres', 'onreso', 'nresou', 'resour', 'esourc', 'source', 'ources', 'urcesg', 'rcesge', 'cesget', 'esgets',
'sgetst', 'getstr', 'etstri', 'tstrin', 'string', 'tringa', 'ringar', 'ingarr', 'ngarra', 'garray', 'arrayr', 'rrayra',
'rayrar', 'ayrarr', 'yrarra', 'rarray', 'arrayl', 'rraylo', 'rayloc', 'ayloca', 'ylocat', 'locati', 'ocatio', 'cation',
'ationd', 'tionda', 'iondat', 'ondata', 'ndatar', 'datare', 'atarep', 'tarepo', 'arepos', 'reposi', 'eposit', 'posito',
'ositor', 'sitory', 'itoryr', 'toryre', 'oryres', 'ryreso', 'yresou', 'resour', 'esourc', 'source', 'ources', 'urcesg',
'rcesge', 'cesget', 'esgets', 'sgetst', 'getstr', 'etstri', 'tstrin', 'string', 'tringa', 'ringar', 'ingarr', 'ngarra',
'garray', 'arrayr', 'rrayra', 'rayrar', 'ayrarr', 'yrarra', 'rarray', 'arrayr', 'rrayre', 'rayrep', 'ayrepo', 'yrepos',
'reposi', 'eposit', 'posito', 'ositor', 'sitory', 'itoryd', 'toryda', 'orydat', 'rydata', 'ydatac', 'dataco', 'atacom',
'tacomp', 'acompa', 'compan', 'ompany', 'mpanyr', 'panyre', 'anyres', 'nyreso', 'yresou', 'resour', 'esourc', 'source',
'ources', 'urcesg', 'rcesge', 'cesget', 'esgets', 'sgetst', 'getstr', 'etstri', 'tstrin', 'string', 'tringa', 'ringar',
'ingarr', 'ngarra', 'garray', 'arrayr', 'rrayra', 'rayrar', 'ayrarr', 'yrarra', 'rarray', 'arrayc', 'rrayco', 'raycom',
'aycomp', 'ycompa', 'compan', 'ompany', 'mpanyd', 'panyda', 'anydat', 'nydata', 'ydataf', 'datafo', 'atafol', 'tafoll',
'afollo', 'follow', 'ollowe', 'llower', 'lowers', 'owersr', 'wersre', 'ersres', 'rsreso', 'sresou', 'resour', 'esourc',
'source', 'ources', 'urcesg', 'rcesge', 'cesget', 'esgets', 'sgetst', 'getstr', 'etstri', 'tstrin', 'string', 'tringa',
```

```
'ringar', 'ingarr', 'ngarra', 'garray', 'arrayr', 'rrayra', 'rayrar', 'ayrarr', 'yrarra', 'rarray', 'arrayf', 'rrayfo',
'rayfol', 'ayfoll', 'yfollo', 'follow', 'ollowe', 'llower', 'lowers', 'owersd', 'wersda', 'ersdat', 'rsdata', 'sdataf',
'datafo', 'atafol', 'tafoll', 'afollo', 'follow', 'ollowi', 'llowin', 'lowing', 'owingr', 'wingre', 'ingres', 'ngreso',
'gresou', 'resour', 'esourc', 'source', 'ources', 'urcesg', 'rcesge', 'cesget', 'esgets', 'sgetst', 'getstr', 'etstri',
'tstrin', 'string', 'tringa', 'ringar', 'ingarr', 'ngarra', 'garray', 'arrayr', 'rrayra', 'rayrar', 'ayrarr', 'yrarra',
'rarray', 'arrayf', 'rrayfo', 'rayfol', 'ayfoll', 'yfollo', 'follow', 'ollowi', 'llowin', 'lowing', 'owingd', 'wingda',
'ingdat', 'ngdata', 'gdataa', 'dataav', 'ataava', 'taavat', 'aavata', 'avatar', 'vatarr', 'atarre', 'tarres', 'arreso',
'rresou', 'resour', 'esourc', 'source', 'ources', 'urceso', 'rcesob', 'cesobt', 'esobta', 'sobtai', 'obtain', 'btaint',
'tainty', 'aintyp', 'intype', 'ntyped', 'typeda', 'ypedar', 'pedarr', 'edarra', 'darray', 'arrayr', 'rrayra', 'rayrar',
'ayrarr', 'yrarra', 'rarray', 'arraya', 'rrayav', 'rayava', 'ayavat', 'yavata', 'avatar', 'vatarp', 'atarpr', 'tarpri',
'arpriv', 'rpriva', 'privat', 'rivate', 'ivatef', 'vatefu', 'atefun', 'tefuns', 'efunsh', 'funsho', 'unshow', 'nshowr',
'showre', 'howrec', 'owrecy', 'wrecyc', 'recycl', 'ecycle', 'cycler', 'yclerv', 'clervi', 'lervie', 'erview', 'rviewv',
'viewva', 'iewval', 'ewvall', 'wvalla', 'vallay', 'allayo', 'llayou', 'layout', 'ayoutm', 'youtma', 'outman', 'utmana',
'tmanag', 'manage', 'anager', 'nagerl', 'agerli', 'gerlin', 'erline', 'rlinea', 'linear', 'inearl', 'nearla', 'earlay',
'arlayo', 'rlayou', 'layout', 'ayoutm', 'youtma', 'outman', 'utmana', 'tmanag', 'manage', 'anager', 'nagert', 'agerth',
'gerthi', 'erthis', 'rthisl', 'thisli', 'hislis', 'islist', 'slistv', 'listvi', 'istvie', 'stview', 'tviewu', 'viewus',
'iewuse', 'ewuser', 'wuser?', 'user?s', 'ser?s ', 'er?s e', 'r?s et', '?s etl', 's etla', ' etlay', 'tlayou', 'layout',
'ayoutm', 'youtma', 'outman', 'utmana', 'tmanag', 'manage', 'anager', 'nagerl', 'agerla', 'gerlay', 'erlayo', 'rlayou',
'layout', 'ayoutm', 'youtma', 'outman', 'utmana', 'tmanag', 'manage', 'anager', 'nagerl', 'agerli', 'gerlis', 'erlist',
'rlistv', 'listvi', 'istvie', 'stview', 'tviewu', 'viewus', 'iewuse', 'ewuser', 'wuser?', 'user?s', 'ser?s ', 'er?s e',
'r?s et', '?s eth', 's etha', ' ethas', 'thasfi', 'hasfix', 'asfixe', 'sfixed', 'fixeds', 'ixedsi', 'xedsiz', 'edsize',
'dsizet', 'sizetr', 'izetru', 'zetrue', 'etrue ', 'true /', 'rue //', 'ue //t', 'e //th', ' //thi', '/thism', 'thisme',
'hismet', 'ismeth', 'smetho', 'method', 'ethoda', 'thodar', 'hodare', 'odaren', 'darene', 'arenee', 'reneed', 'eneedt',
'needto', 'eedtom', 'edtoma', 'dtomak', 'tomake', 'omakes', 'makesa', 'akesam', 'kesame', 'esameo', 'sameor', 'ameord',
'meorde', 'eorder', 'orderl', 'rderli', 'derlik', 'erlike', 'rliket', 'liketh', 'ikethe', 'kethed', 'etheda', 'thedat',
'hedata', 'edatap', 'datapr', 'atapri', 'tapriv', 'apriva', 'privat', 'rivate', 'ivatef', 'vatefu', 'atefun', 'tefuna',
'efunad', 'funadd', 'unaddd', 'naddda', 'adddat', 'dddata', 'ddataf', 'datafo', 'atafor', 'taforp', 'aforpo', 'forpos',
'orposi', 'rposit', 'positi', 'ositio', 'sition', 'itioni', 'tionin', 'ionind', 'oninda', 'nindat', 'indata', 'ndatan',
'datana', 'atanam', 'taname', 'anamei', 'namein', 'ameind', 'meindi', 'eindic', 'indice', 'ndices', 'dicesv', 'icesva',
'cesval', 'esvalu', 'svalus', 'valuse', 'aluser', 'luseru', 'userus', 'seruse', 'eruser', 'ruserd', 'userda', 'serdat',
'erdata', 'rdatau', 'dataus', 'atause', 'tauser', 'ausern', 'userna', 'sernam', 'ername', 'rnamep', 'namepo', 'amepos',
'meposi', 'eposit', 'positi', 'ositio', 'sition', 'itiond', 'tionda', 'iondat', 'ondata', 'ndatan', 'datana', 'atanam',
'taname', 'anamep', 'namepo', 'amepos', 'meposi', 'eposit', 'positi', 'ositio', 'sition', 'itiond', 'tionda', 'iondat',
'ondata', 'ndatal', 'datalo', 'ataloc', 'taloca', 'alocat', 'locati', 'ocatio', 'cation', 'ationp', 'tionpo', 'ionpos',
'onposi', 'nposit', 'positi', 'ositio', 'sition', 'itiond', 'tionda', 'iondat', 'ondata', 'ndatar', 'datare', 'atarep',
'tarepo', 'arepos', 'reposi', 'eposit', 'posito', 'ositor', 'sitory', 'itoryp', 'torypo', 'orypos', 'ryposi', 'yposit',
'positi', 'ositio', 'sition', 'itiond', 'tionda', 'iondat', 'ondata', 'ndatac', 'dataco', 'atacom', 'tacomp', 'acompa',
'compan', 'ompany', 'mpanyp', 'panypo', 'anypos', 'nyposi', 'yposit', 'positi', 'ositio', 'sition', 'itiond', 'tionda',
'iondat', 'ondata', 'ndataf', 'datafo', 'atafol', 'tafoll', 'afollo', 'follow', 'ollowe', 'llower', 'lowers', 'owersp',
'werspo', 'erspos', 'rsposi', 'sposit', 'positi', 'ositio', 'sition', 'itiond', 'tionda', 'iondat', 'ondata', 'ndataf',
'datafo', 'atafol', 'tafoll', 'afollo', 'follow', 'ollowi', 'llowin', 'lowing', 'owingp', 'wingpo', 'ingpos', 'ngposi',
'gposit', 'positi', 'ositio', 'sition', 'itiond', 'tionda', 'iondat', 'ondata', 'ndataa', 'dataav', 'ataava', 'taavat',
'aavata', 'avatar', 'vatarg', 'atarge', 'target', 'argetr', 'rgetre', 'getres', 'etreso', 'tresou', 'resour', 'esourc',
'source', 'ourcei', 'urceid', 'rceidp', 'ceidpo', 'eidpos', 'idposi', 'dposit', 'positi', 'ositio', 'sition', 'ition-',
'tion-1', 'ion-1u', 'on-1us', 'n-1use', '-1user', '1users', 'usersa', 'sersad', 'ersadd', 'rsaddu', 'saddus', 'adduse',
'dduser', 'dusera', 'userad', 'serada', 'eradap', 'radapt', 'adapte', 'dapter', 'apterl', 'pterli', 'terlis', 'erlist',
'rlistu', 'listus', 'istuse', 'stuser', 'tuseru', 'userus', 'seruse', 'eruser', 'rusers', 'users/', 'sers//', 'ers//e',
'rs//en', 's//end', '//end ', '/end \n']
```

3. Calculation of the hash function per k-gram

The third step is to formulate a hash function calculation, each k-gram is made rolling hash, along with testing the k-gram hash calculation on source code 1 and 2. The source code to calculate of the hash function per k-gram on Figure 3.3

```python
def rolling_hash(self,text):
    '''
    rolling_hash is another means of generating a numerical representation of a document (if you choose not to
    go with the default). This function will return the hashes for all k-grams in o(n) and can be used if
    computation complexity is a problem
    Parameters
    ----------
    text: str
        The entire document to be generate hashed k-grams
    Returns
    -------
    hashes: list of int
        A list of all hashed k-grams
    '''
    hashes= []
    E = 10 **(self.k - 1)
    text = str(text)
    next_str = text[0:self.k]
    next_hash = self.horners_rule(next_str)
    hashes.append(next_hash)
    for i in range(0, len(text) - self.k):
        current_str = next_str
        current_hash = next_hash
        if text[i] != " ":
            # get the new stuff
            next_str = text[i + 1: self.k + i + 1]
            next_hash = ((current_hash - (ord(current_str[0]) * E)) * 10) + ord(next_str[self.k - 1])
            # hashes.append((next_hash, i + 1, next_str))
            hashes.append(next_hash)
            # print("Window:", next_str)
            # print("Hash:", next_hash)
    return hashes
```

Figure 3.3 Source code to calculate of the hash function per k-gram

Source code 1

```
[12280773, 10807831, 11078409, 11784201, 10842119, 11421305, 11213167, 11131768, 12317789, 12177995, 12780065, 12800765,
11007755, 12077661, 11776720, 12767303, 12673135, 11731466, 12314764, 12147757, 11477668, 11776797, 12768085, 11680951,
12809624, 11096337, 12963482, 12634932, 11349428, 12494385, 10943949, 12439587, 12395986, 11959965, 11599761, 10997720,
10977305, 12773159, 11731702, 12317131, 12171424, 11714356, 12143657, 12436680, 12366900, 12669114, 12691251, 10912615,
12126250, 11262599, 12626101, 12261120, 11611316, 11113261, 11132720, 12327316, 12273265, 12732760, 11327716, 12277261,
12772720, 11727316, 12273265, 12732759, 11327702, 12277131, 12771424, 11714356, 12143657, 12436680, 12366900, 12669114,
12691251, 10912615, 12126250, 11262599, 12626101, 12261120, 11611316, 11113261, 11132720, 12327316, 12273274, 12732841,
11328525, 12285366, 12853781, 12537922, 11379321, 12793310, 12933197, 13332084, 12320954, 11209637, 11096491, 10965015,
12650259, 12502702, 11027131, 13271424, 11714356, 12143657, 12436680, 12366900, 12669114, 12691251, 10912615, 12126250,
11262611, 12626225, 12262348, 11623597, 11236080, 12360900, 12609108, 11091181, 12911915, 12119259, 11192702, 11927131,
11271424, 11714356, 12143657, 12436680, 12366900, 12669114, 12691251, 10912615, 12126250, 11262619, 12626295, 12263050,
11630603, 11306131, 13061426, 11614357, 11143670, 11436797, 11368082, 12680936, 10809461, 11094724, 10947358, 12473685,
12736951, 11369629, 12696395, 12964059, 11640702, 11407131, 13071424, 11714356, 12143657, 12436680, 12366900, 12669114,
12691251, 10912615, 12126250, 11262619, 12626295, 12263050, 11630603, 11306131, 13061426, 11614368, 11143785, 11437965,
11379766, 12797778, 11977885, 11778951, 12789629, 12896395, 12964059, 11640702, 11407131, 13071424, 11714356, 12143657,
12436680, 12366900, 12669114, 12691251, 10912615, 12126250, 11262620, 12626297, 12263082, 11630932, 11309419, 13094301,
10943119, 12431302, 12313117, 11131286, 12312957, 12129682, 12296932, 10969417, 12694282, 10942932, 12429419, 12294301,
10943119, 12431302, 12313117, 11131286, 12312957, 12129669, 12296806, 10968165, 12681768, 10817785, 11177966, 12779781,
11797909, 12979198, 11792077, 12920885, 13208965, 11089759, 11897687, 10976975, 12769860, 12698697, 11987069, 10870806,
11708165, 12081768, 10817785, 11177966, 12779781, 11797907, 12979182, 11791932, 12919419, 13194301, 10943119, 12431302,
12313117, 11131286, 12312957, 12129669, 12296806, 10968165, 12681768, 10817785, 11177966, 12779781, 11797922, 12979334,
11793445, 12934568, 13345777, 12457886, 12578961, 11789718, 12897277, 10972886, 12728961, 11289715, 11897260, 10972705,
12727166, 11271778, 11717877, 12178884, 11788937, 12889470, 12894797, 10948082, 12480936, 10809461, 11094724, 10947357,
12473685, 12736951, 11369624, 12696337, 12963470, 12634797, 11348082, 12480936, 10809461, 11094724, 10947352, 12473634,
12736445, 11364568, 12645777, 12457886, 12578961, 11789718, 12897277, 10972886, 12728961, 11289715, 11897260, 10972705,
12727166, 11271778, 11717877, 12178884, 11788940, 12889497, 12895086, 10950957, 12509687, 11096985, 10969951, 12699624,
10996350, 12963597, 12636079, 11360891, 12609007, 12090184, 10901954, 12019637, 11196491, 10965025, 12650366, 12503774,
11037845, 13378560, 12785703, 12857142, 12571534, 11715445, 12154568, 11545777, 12457886, 12578961, 11789718, 12897277,
10972886, 12728961, 11289715, 11897260, 10972705, 12727166, 11271778, 11717877, 12178884, 11788940, 12889497, 12895086,
10950957, 12509680, 11096897, 10969079, 12690891, 10909007, 12090184, 10901954, 12019637, 11196491, 10965025, 12650366,
12503774, 11037845, 13378560, 12785703, 12857142, 12571534, 11715445, 12154568, 11545777, 12457886, 12578961, 11789718,
12897277, 10972886, 12728961, 11289715, 11897260, 10972705, 12727166, 11271778, 11717877, 12178884, 11788940, 12889497,
12895086, 10950957, 12509669, 11096801, 10968119, 12681302, 10813117, 11131280, 12312921, 12129307, 12293184, 10931954,
12319637, 13196491, 10965025, 12650366, 12503774, 11037845, 13378560, 12785703, 12857142, 12571534, 11715445, 12154568,
11545777, 12457886, 12578961, 11789718, 12897277, 10972886, 12728961, 11289715, 11897260, 10972705, 12727166, 11271778,
11717877, 12178884, 11788940, 12889497, 12895086, 10950957, 12509682, 11096924, 10969351, 12693626, 10936371, 12363826,
11638381, 12383922, 12839321, 12393310, 12933197, 13332084, 12320954, 11209637, 11096491, 10965022, 12650334, 12503445,
11034568, 13345777, 12457886, 12578961, 11789718, 12897277, 10972886, 12728961, 11289715, 11897260, 10972705, 12727166,
11271778, 11717877, 12178884, 11788940, 12889497, 12895086, 10950957, 12509678, 11096891, 10969009, 12690187, 10901986,
12019965, 12199761, 10997720, 10977297, 12773084, 11730954, 12309637, 12096491, 10965025, 12650366, 12503774, 11037845,
13378560, 12785703, 12857142, 12571534, 11715445, 12154568, 11545777, 12457886, 12578961, 11789718, 12897277, 10972886,
12728961, 11289715, 11897260, 10972705, 12727166, 11271778, 11717877, 12178884, 11788940, 12889497, 12895086, 10950957,
12509684, 11096941, 10969522, 12695331, 10953425, 12534355, 11343666, 12436771, 12367824, 12678361, 11783707, 12837184,
12371954, 12719637, 13196491, 10965025, 12650366, 12503774, 11037845, 13378560, 12785703, 12857142, 12571534, 11715445,
12154568, 11545777, 12457886, 12578961, 11789718, 12897277, 10972886, 12728961, 11289715, 11897260, 10972705, 12727166,
11271778, 11717877, 12178884, 11788940, 12889497, 12895086, 10950957, 12509672, 11096831, 10968418, 12684288, 10842991,
11430029, 12300391, 12004024, 12040355, 12403647, 13036584, 11365954, 12659637, 12596491, 10965025, 12650366, 12503774,
11037845, 13378560, 12785703, 12857142, 12571534, 11715445, 12154568, 11545777, 12457886, 12578961, 11789718, 12897277,
10972886, 12728961, 11289715, 11897260, 10972705, 12727166, 11271778, 11717877, 12178884, 11788940, 12889497, 12895086,
10950957, 12509672, 11096831, 10968418, 12684288, 10842991, 11430029, 12300395, 12004060, 12040703, 12407127, 13071384,
11713954, 12139637, 11396491, 10965025, 12650366, 12503774, 11037845, 13378560, 12785703, 12857142, 12571534, 11715445,
12154568, 11545777, 12457886, 12578961, 11789728, 12897377, 10973884, 12738957, 11389685, 12896951, 10969624, 12696355,
12963647, 12636584, 11365954, 12659637, 12596491, 10965018, 12650285, 12502965, 11029766, 13297771, 11977812, 11778237,
12782485, 12824951, 12249624, 11496351, 12963628, 12636381, 11363924, 12639354, 12393645, 12936550, 11365601, 12656112,
12561237, 11612480, 11124911, 11249220, 11492299, 12923104, 12231141, 12311507, 12115186, 11151961, 12519725, 11197347,
10973588, 12735981, 11359910, 12599205, 10992160, 12921715, 11217266, 11172757, 11727680, 12276899, 12769091, 12691025,
10910366, 12103757, 11037686, 11376961, 12769708, 12697197, 10972080, 12720900, 11209108, 11091181, 12911873, 12118845,
11188482, 11884937, 10849482, 7494921, 11949324, 4493351, 12933609, 17336204, 72362141, 609621507, 5985215186,
59742151961, 597322519725, 5973111197347, 59731010973588, 597310012735981, 5973100011359910, 59731000012599205,
597310000010992160, 5973100000012921715, 59731000000011217266, 597310000000011172757, 5973100000000011727680,
597310000000000012276899, 5973100000000000012769091, 59731000000000000012691025, 597310000000000000010910366]
```

Source code 2

```
[12280773, 10807831, 11078409, 11784201, 10842119, 11421299, 11213087, 11130985, 12309967, 12099789, 11997991, 10980025,
12800353, 13003635, 13036466, 11364764, 12647757, 11477668, 11776797, 12768085, 11680951, 12809624, 11096342, 12963525,
12635364, 11353755, 12537666, 11376765, 11767759, 12677702, 12777131, 12771424, 11714356, 12143657, 12436680, 12366900,
12669114, 12691251, 10912615, 12126250, 11262599, 12626101, 12261120, 11611316, 11113261, 11132720, 12327316, 12273274,
12732841, 11328525, 12285366, 12853781, 12537922, 11379321, 12793310, 12933197, 13332084, 12320954, 11209637, 11096491,
10965015, 12650259, 12502702, 11027131, 13271424, 11714356, 12143657, 12436680, 12366900, 12669114, 12691251, 10912615,
12126250, 11262620, 12626297, 12263082, 11630932, 11309419, 13094301, 10943119, 12431302, 12313117, 11131286, 12312957,
12129682, 12296932, 10969417, 12694282, 10942932, 12429419, 12294301, 10943119, 12431302, 12313117, 11131286, 12312957,
12129669, 12296806, 10968165, 12681768, 10817785, 11177966, 12779781, 11797915, 12979259, 11792702, 12927131, 13271424,
11714356, 12143657, 12436680, 12366900, 12669114, 12691251, 10912615, 12126250, 11262611, 12626225, 12262348, 11623597,
11236080, 12360900, 12609108, 11091181, 12911915, 12119259, 11192702, 11927131, 11271424, 11714356, 12143657, 12436680,
12366900, 12669114, 12691251, 10912615, 12126250, 11262620, 12626314, 12263241, 11632509, 11325211, 13252209, 12522198,
11222081, 11220924, 13209358, 11093685, 11936951, 11369629, 12696409, 12964195, 11642050, 11420603, 13206131, 13061426,
11614368, 11143785, 11437960, 11379701, 12797107, 11971184, 11711948, 12119577, 11195891, 10959021, 12590327, 11903386,
11033969, 13339787, 12397980, 12979897, 12799073, 11990831, 10908424, 12084345, 10843559, 11435702, 11357131, 12571424,
11714356, 12143657, 12436680, 12366900, 12669114, 12691251, 10912615, 12126250, 11262620, 12626314, 12263241, 11632509,
11325211, 13252209, 12522198, 11222081, 11220924, 13209358, 11093685, 11936951, 11369629, 12696409, 12964195, 11642050,
11420603, 13206131, 13061426, 11614374, 11143841, 11438509, 11385211, 12852209, 12522198, 11222081, 11220924, 13209358,
11093685, 11936951, 11369629, 12696395, 12964059, 11640702, 11407131, 13071424, 11714356, 12143667, 12436781, 12367926,
12679368, 12793785, 11937960, 12379720, 12797297, 11973080, 11730900, 12309114, 13091251, 10912615, 12126250, 11262615,
12626271, 12262820, 11628316, 11283264, 12832741, 13327526, 12275365, 12753749, 11537599, 11376087, 12760975, 11609860,
11098697, 11987069, 10870806, 11708165, 12081768, 10817785, 11177966, 12779781, 11797919, 12979287, 11792975, 12929860,
13298632, 11986362, 10863652, 11636619, 11366298, 3663077, 4630885, 14308947, 35089575, 253895860, 2423958697,
24124587069, 241136870806, 2411271708165, 24112612081768, 241126010817785, 2411260011177966, 24112600012779781,
241126000011797907, 2411260000012979182, 24112600000011791932, 241126000000012919419, 2411260000000013194301]
```

4. Formation of window values from hash values

Do grouping (Windowing) for each hash result, the steps are the same as with k-grams. Following are some of the results of the window series in source code 1 and 2 Source code 1. The source code to create formation of window values from hash values on Figure

```
def select_fingerprints(self, hash_list, w):
    '''
    This helper function takes in a  set of
    Parameters
    ----------
    hash_list : list of (int, int)

    w : int
        w represents the window size for which we select a rightmost minimum

    Returns
    -------
    fingerprints : list of int
    '''
    fingerprints = []
    min_index = -1
    prev_min_index = -1
    # traverse over the hash_list
    for hash_index in range(len(hash_list) - w +1):
        min_value =  float("inf")
        #traverse over each window
        for window_index in range(hash_index, hash_index + w ):
            if hash_list[window_index] <= min_value:
                min_index = window_index
                min_value = hash_list[window_index]
        # If the minimum value of the previous window is no longer the minimum value
        if min_index != prev_min_index:
          prev_min_index = min_index
          fingerprints.append(hash_list[min_index])
```

Figure 3.4 Source code to create formation of window values

```
[122797, 108067, 110773]
[108067, 110773, 117831]
[110773, 117831, 108409]
[117831, 108409, 114201]
[108409, 114201, 112119]
[114201, 112119, 111305]
[112119, 111305, 123167]
[111305, 123167, 121768]
[123167, 121768, 127789]
[121768, 127789, 127995]
[127789, 127995, 110065]
[127995, 110065, 120765]
[110065, 120765, 117755]
[120765, 117755, 127661]
[117755, 127661, 126720]
[127661, 126720, 117303]
[126720, 117303, 123135]
[117303, 123135, 121466]
[123135, 121466, 114764]
[121466, 114764, 117757]
[114764, 117757, 127668]
[117757, 127668, 116797]
[127668, 116797, 128085]
[116797, 128085, 110951]
[128085, 110951, 129624]
[110951, 129624, 126337]
[129624, 126337, 113482]
[126337, 113482, 124932]
[113482, 124932, 109428]
[124932, 109428, 124385]
[109428, 124385, 123949]
[124385, 123949, 119587]
[123949, 119587, 115986]
[119587, 115986, 109965]
[115986, 109965, 109761]
[109965, 109761, 127720]
[109761, 127720, 117305]
[127720, 117305, 123159]
[117305, 123159, 121702]
[123159, 121702, 117131]
[121702, 117131, 121424]
[117131, 121424, 124356]
[121424, 124356, 123657]
[124356, 123657, 126680]
[123657, 126680, 126900]
[126680, 126900, 109114]
[126900, 109114, 121251]
[109114, 121251, 112615]
[121251, 112615, 126250]
[112615, 126250, 122599]
[126250, 122599, 116101]
[122599, 116101, 111120]
```

Source code 2

```
[122797, 108067, 110773]
[108067, 110773, 117831]
[110773, 117831, 108409]
[117831, 108409, 114201]
[108409, 114201, 112119]
[114201, 112119, 111299]
[112119, 111299, 123087]
[111299, 123087, 120985]
[123087, 120985, 119967]
[120985, 119967, 109789]
[119967, 109789, 127991]
[109789, 127991, 130025]
[127991, 130025, 130353]
[130025, 130353, 113635]
[130353, 113635, 126466]
[113635, 126466, 114764]
[126466, 114764, 117757]
[114764, 117757, 127668]
[117757, 127668, 116797]
[127668, 116797, 128085]
[116797, 128085, 110951]
[128085, 110951, 129624]
[110951, 129624, 126342]
[129624, 126342, 113525]
[126342, 113525, 125364]
[113525, 125364, 113755]
[125364, 113755, 117666]
[113755, 117666, 126765]
[117666, 126765, 127759]
[126765, 127759, 127702]
[127759, 127702, 117131]
[127702, 117131, 121424]
[117131, 121424, 124356]
[121424, 124356, 123657]
[124356, 123657, 126680]
[123657, 126680, 126900]
[126680, 126900, 109114]
[126900, 109114, 121251]
[109114, 121251, 112615]
[121251, 112615, 126250]
[112615, 126250, 122599]
[126250, 122599, 116101]
[122599, 116101, 111120]
[116101, 111120, 111316]
[111120, 111316, 123261]
[111316, 123261, 122720]
[123261, 122720, 127316]
[122720, 127316, 113274]
[127316, 113274, 122841]
[113274, 122841, 128525]
[122841, 128525, 125366]
[128525, 125366, 113781]
[125366, 113781, 127922]
[113781, 127922, 129321]
[127922, 129321, 133310]
[129321, 133310, 123197]
```

5. Fingerprint selection of each window

In this step, take the smallest hash value from the Window series which is called the Fingerprint. The following is the result of fingerprint selection in source code 1 and 2. The source code to select fingerprint of each window on Figure 3.5

```
def generate_fingerprints(doc_path):
    '''
    Given the path of a text file, this function generates fingerprints for the contents of the file.
    Parameters
    ----------
    doc_path : str
        path to the text file
    Returns
    -------
    fingerprints : list of str
        list of fingerprints for the given document
    '''
    with open(doc_path) as document:
        contents = document.read()
        # print(contents)
        # create a new instance of the WinnowedDoc class with the updated constructor
        winnowed_doc = WinnowedDoc(contents, k_gram_size, window_size, True)


        # call the fingerprints method on the new instance
        fingerprints = winnowed_doc.fingerprints
    return fingerprints
```

Figure 3.5 Source code to select fingerprint of each window

Source code 1

```
[108067, 108409, 111305, 121768, 110065, 117755, 117303, 114764, 116797, 110951, 113482, 109428, 119587, 115986, 109965,
109761, 117305, 117131, 121424, 123657, 109114, 112615, 116101, 111120, 111316, 122720, 113265, 117261, 113265, 117131,
121424, 123657, 109114, 112615, 116101, 111120, 111316, 122720, 113274, 122841, 113781, 127922, 123197, 112084, 110954,
109637, 110259, 117131, 121424, 123657, 109114, 112615, 116225, 112348, 110900, 111915, 112702, 117131, 121424, 123657,
109114, 112615, 116295, 113050, 111426, 113670, 108082, 109461, 113685, 116395, 114059, 117131, 121424, 123657, 109114,
112615, 116295, 113050, 111426, 113785, 117778, 127885, 116395, 114059, 117131, 121424, 123657, 109114, 112615, 116297,
113082, 109419, 111302, 121286, 109682, 109417, 109419, 111302, 121286, 109669, 108165, 111768, 117966, 117909, 110885,
109759, 119860, 108697, 108165, 111768, 117966, 117907, 109419, 111302, 121286, 109669, 108165, 111768, 117966, 117922,
124568, 117886, 109718, 112886, 109715, 112705, 117166, 117877, 109470, 108082, 109461, 113685, 126337, 113470, 108082,
109461, 113634, 124568, 117886, 109718, 112886, 109715, 112705, 117166, 117877, 109497, 109687, 109951, 113597, 109007,
109637, 110366, 127845, 125703, 117142, 115445, 117886, 109718, 112886, 109715, 112705, 117166, 117877, 109497, 109680,
109079, 109007, 109637, 110366, 127845, 125703, 117142, 115445, 117886, 109718, 112886, 109715, 112705, 117166, 117877,
109497, 109669, 108119, 111302, 121280, 109307, 109637, 110366, 127845, 125703, 117142, 115445, 117886, 109718, 112886,
109715, 112705, 117166, 117877, 109497, 109682, 109351, 116371, 123826, 123922, 123197, 112084, 110954, 109637, 110334,
124568, 117886, 109718, 112886, 109715, 112705, 117166, 117877, 109497, 109678, 109009, 109965, 109761, 117297, 109637,
110366, 127845, 125703, 117142, 115445, 117886, 109718, 112886, 109715, 112705, 117166, 117877, 109497, 109684, 109522,
113425, 123666, 117824, 123707, 109637, 110366, 127845, 125703, 117142, 115445, 117886, 109718, 112886, 109715, 112705,
117166, 117877, 109497, 109672, 108418, 114288, 120029, 120391, 113647, 109637, 110366, 127845, 125703, 117142, 115445,
117886, 109718, 112886, 109715, 112705, 117166, 117877, 109497, 109672, 108418, 114288, 120029, 120395, 117127, 113954,
109637, 110366, 127845, 125703, 117142, 115445, 117886, 109728, 113884, 109685, 126355, 113647, 109637, 110285, 117771,
122485, 114951, 113628, 123924, 113645, 116112, 111237, 112480, 114911, 122299, 121141, 111507, 109725, 113588, 109910,
111715, 117266, 122757, 109091, 110366, 113757, 109708, 110900, 111873, 108482, 74937, 44921, 129321, 173320, 723299,
6093104, 59821141, 597111507, 5970125186, 59700111961, 597000109725, 5970000127347, 59700000113588, 597000000125981,
5970000000109910, 59700000000129205, 597000000000112160, 5970000000000111715]
```

Source code 2

```
[108067, 108409, 111299, 119967, 109789, 127991, 113635, 114764, 116797, 110951, 113525, 113755, 117666, 126765, 117131,
121424, 123657, 109114, 112615, 116101, 111120, 111316, 122720, 113274, 122841, 113781, 127922, 123197, 112084, 110954,
109637, 110259, 117131, 121424, 123657, 109114, 112615, 116297, 113082, 109419, 111302, 121286, 109682, 109417, 109419,
111302, 121286, 109669, 108165, 111768, 117966, 117915, 117131, 121424, 123657, 109114, 112615, 116225, 112348, 110900,
111915, 112702, 117131, 121424, 123657, 109114, 112615, 116314, 113241, 112209, 112198, 110924, 113685, 116409, 114195,
116131, 111426, 113785, 117107, 111948, 109577, 110327, 123969, 119897, 109073, 108424, 113559, 117131, 121424, 123657,
109114, 112615, 116314, 113241, 112209, 112198, 110924, 113685, 116409, 114195, 116131, 111426, 113841, 112209, 112198,
110924, 113685, 116395, 114059, 117131, 121424, 123667, 119368, 119720, 117297, 109114, 112615, 116271, 112820, 122741,
115365, 113749, 110975, 108697, 108165, 111768, 117966, 117919, 119860, 108632, 36619, 46287, 142985, 349959, 2529687,
24146975, 240319860, 2402108697, 24020117069, 240200120806, 2402000108165, 24020000111768, 240200000127785,
2402000000117966, 24020000000129781, 240200000000117907, 2402000000000129182, 24020000000000131932,
240200000000000109419, 2402000000000000012430]
```

Table 1 explains the effect of n-grams and w-grams on the similarity results. The testing results on two samples or documents are in table 1 with a value of *k-gram* = 4 and *w-gram* = 3. The reason for choosing the value of k = 4 is because the greater the value of k, the greater the similarity of values obtained. The use of preprocessing greatly affects the similarity value. Table 1, for the same source code, has a 100% similarity value and can be considered the proposed method to be successful because there is no difference, so the k-gram values are the same and also produce the same hash value. In table 1, the similarity value using preprocessing is lower than without preprocessing. This can happen because the

number of characters created by the data that has been preprocessed is less. Tests in table 1 numbers 1, 2, and 3 produce high similarity. Lower than the test in number 1. The k-gram and w-gram values also influence this because the smaller the k-gram and w-gram values, the more often the pieces will be matched and usually found. Otherwise, the seldom is the data compared or found.

The applied preprocessing (removing some standard components and grammar tokenization) has the benefit of increasing the performance of metrics for calculating fingerprints in this study and research [8]. Using preprocessing tokenization with grammar will speed up the fingerprint calculation process because fewer letters are processed. Without preprocessing, it will cause parts often repeated/ templates in a code to have a more dominating effect than the changes made. The use of stemming affects the accuracy of the resulting similarity value. Using stemming produces fewer good scores than those without stemming [3].

The Winnowing algorithm has *k-grams*, and the window process can be changed. The higher the *k-gram* value, the lower the similarity value is produced, and the lower the k-gram value, the higher the similarity value.

But that doesn't mean a low *k-gram* value will give an accurate accuracy value. The smaller the *k-gram* value, the smaller the characters to match and the more often these characters are found in the text.

Table 1. Plagiarism Detection Trial Using Different Datasets Against Similarity Results with the Jaccard Similarity Method

No	*Input*	*Preprocessing*	*K-Gram*	*W-Gram*	*Similarity*
1.	Precisely the same source code	Yes	4	3	100%
2.	The source code is nearly the same	Yes	4	3	100%
3.	Source code that has a big difference	Yes	4	3	3,92%
4.	Precisely the same source code	No	4	3	100%
5.	The source code is nearly the same	No	4	3	0,19%
6.	Source code that has a big difference	No	4	3	7,68%

The following is an example of the results of source code detection, and the author adds a threshold variable that functions to provide a minimum value to determine whether the source code is indicated as plagiarism or not.

```
Enter the path to the first text file: /content/gdrive/MyDrive/Dataset The
Enter the path to the second text file: /content/gdrive/MyDrive/Dataset Th
Enter the similarity threshold (0-100): 70
The two files are similar with a Jaccard similarity of 3.926347143243975%
```

Figure 2. Examples of implementation results to detect plagiarism

4. Conclusion

From the research results above, it can be understood that text preprocessing greatly influences similarity results. The degree of similarity in the two source codes will produce different similarity values if the dataset used has gone through the text preprocessing stage or without preprocessing. If the dataset has gone through the text preprocessing step, the similarity value will be low because the number of characters used is significantly reduced. When compared, the value of similarity by the dataset without using preprocessing is higher than the dataset that has been done preprocessing.

The k-gram and w-gram values also influence this because the smaller the k-gram and w-gram values, the more often the pieces will be matched and found frequently. Otherwise, the data will be checked or seen more rarely.

The Jaccard similarity, which is used to produce similarity levels between source codes, is also quite good because this coefficient is simple by finding the same item from two documents and then dividing by the total of the second item from combining the two source codes.

The winnowing algorithm detects source code well and quickly, so it can be proven that the winnowing algorithm cannot only detect plagiarism in text documents but can also be used for source code. Plagiarism detection on source code using the winnowing algorithm is good enough to use to compare similarities between source code and good enough to use to minimize plagiarism. Future research is expected to be able to detect plagiarism by using other programming languages and implementing better methods in the process of detecting plagiarism.

References

[1] G. Cosma and M. Joy, "Towards a definition of source-code plagiarism," IEEE Transactions on Education, vol. 51, no. 2, pp. 195–200, May 2008, doi: 10.1109/TE.2007.906776.

[2] H. Cheers, Y. Lin, and S. P. Smith, "Academic Source Code Plagiarism Detection by Measuring Program Behavioral Similarity," IEEE Access, vol. 9, pp. 50391–50412, 2021, doi: 10.1109/ACCESS.2021.3069367.

[3] D. Leman, B. S. Riza, and M. B. Akbbar, "RABIN KARP AND WINNOWING ALGORITHM FOR STATISTICS OF TEXT DOCUMENT PLAGIARISM DETECTION," 2019.

[4] E. Y. Puspaningrum, B. Nugroho, A. Setiawan, and N. Hariyanti, "Detection of Text Similarity for Indication Plagiarism Using Winnowing Algorithm Based K-gram and Jaccard Coefficient," in Journal of Physics: Conference Series, IOP Publishing Ltd, Jul. 2020. doi: 10.1088/1742-6596/1569/2/022044.

[5] M. S. Ramli, S. Cokrowibowo, and M. F. Rustan, "Uji Plagiarism pada Tugas Mahasiswa Menggunakan Algoritma Winnowing," Journal of Applied Computer Science and Technology, vol. 2, no. 2, pp. 108–112, Dec. 2021, doi: 10.52158/jacost.v2i2.177.

[6] I. Widaningrum, D. Mustikasari, R. Arifin, and H. A. Pratiwi, "Evaluation of the accuracy of winnowing, rabin karp and knuth morris pratt algorithms in plagiarism detection applications," in Journal of Physics: Conference Series, Institute of Physics Publishing, May 2020. doi: 10.1088/1742- 6596/1517/1/012093.

[7] K. Rinartha, "SIMPLE QUERY SUGGESTION UNTUK PENCARIAN ARTIKEL MENGGUNAKAN JACCARD SIMILARITY," Jurnal Ilmiah Rekayasa dan Manajemen Sistem Informasi, vol. 3, no. 1, 2017.

[8] N. Awale, M. Pandey, A. Dulal, and B. Timsina, "Plagiarism Detection in Programming Assignments using Machine Learning," Journal of Artificial Intelligence and Capsule Networks, vol. 2, no. 3, pp. 177–184, Jul. 2020, doi: 10.36548/jaicn.2020.3.005.

[9] st Muhammad Faisal, I. Malang Malang, rd M. Maulana El Sulthan, th Fauziyah Amini, and th M. Amin Hariyadi, "Plagiarism Detection Using Manber and Winnowing Algorithm," International Journal of Advanced Science and Technology, vol. 29, no. 6s, pp. 2130–2136, 2020.

[10] T. Widjaja and A. Gunawan, "Deteksi Plagiarisme pada Kode Bahasa Pemrograman Java menggunakan XGBoost," 2020.

[11] R. Ramdhani, A. Fadlil, and S. Sunardi, "Penerapan Algoritma Winnowing dan Word-Level Trigrams Untuk Mengidentifikasi Kesamaan Kata," JURIKOM (Jurnal Riset Komputer), vol. 9, no. 2, p. 427, Apr. 2022, doi: 10.30865/jurikom.v9i2.4060.

[12] S. Purwaningrum, A. Susanto, N. Wachid, and A. Prasetya, "Comparation of Dice Similarity and Jaccard Coefficience Against Winnowing Algorithm For Similarity Detection of Indonesian Text Documents," 2021.

[13] S. Schleimer, D. S. Wilkerson, and A. Aiken, "Winnowing: Local Algorithms for Document Fingerprinting," p. 702, 2003.