International Journal of Computer Science and Engineering (IJCSE)
ISSN(P): 2278-9960; ISSN(E): 2278-9979
Vol. 6, Issue 3, Apr - May 2017; 1-20

International Academy of Science, Engineering and Technology
Connecting Researchers; Nurturing Innovations

NEW RULES BASED APPROACH FOR ARABIC TEXT DATA HIDING

MOHAMMED JAWAR KHAMI

Assistant Professor, Department of Technical Computer Systems, Basra Technical Institute,
Southern Technical University, Iraq

ABSTRACT

In steganography, there are many hiding techniques to do the job of data hiding. These techniques differ from each other either by the applied hiding approach or by the used cover object. From the cover object side view, and even though text documents are unavoidable form of information communication among humans, researches on text hiding techniques are less in contrast to other cover object's techniques. This is due to that text documents have relatively less number of features that can be used to hide data in comparison with other cover object types. In this paper, new text hiding approach is proposed, written in algorithms form, and then coded in mat lab (m-files). Text hiding technique used in this work has many advantages over other existing text-in-text hiding techniques. These advantages include the usage of Arabic or Arabic-English mixed cover text with the aids of two of the non-printing Unicode characters. Also applying new hiding rules concerning Arabic writing system. The cover text classified into groups of Arabic letters each with specific features and thus hiding text in between letters from these groups must controlled by these new text hiding rules. Mat lab programs for embedding and extracting the secret text, according to the new approach, are tested and the outputs have been found very satisfying. Both secret and cover text have the same original format and text configuration.

KEYWORDS: Cryptography, Embedding and Extraction Algorithms, Huffman Code System, Unicode Characters Set

INTRODUCTION

Cryptography and steganography are two systems almost with the same goal. Both intend to secure the transfer of digital data over the Internet or throw other communication channels. Cryptography secures data by transforming it into another, unreadable format. While steganography makes **secret** data invisible, by hiding them in another piece of data, known as the **cover** object. The modified cover (including the secret hidden data), is referred to as a **stego** object. It can be stored or transmitted as normal message [1].

In steganography, there are many hiding techniques to do the job. These techniques differ from each other either by the applied hiding approach or by the used cover object. From the used cover object side view, most of today's techniques deal with digital image, audio or video documents. Nonetheless, text documents (either in printed or digital form), are still the most common and almost unavoidable form of information communication among humans [2], techniques of data hiding in text are less in contrast to other cover object's techniques. The main reason for this is that text documents have a relatively less number of features that can be used to hide data in comparison with other cover object types.

Text steganography is believed to be the trickiest due to deficiency of redundant information which is present in image, audio or a video file [3]. The real structure of text character symbols is identical with what we see or observe, while in other types of cover objects, such as in image, video, or audio the real structure of cover objects is different from

what we observe. Therefore, in such cover objects, we can hide information by introducing changes in the structure of the cover document without making a notable change in the concerned output form [4].

In this paper, new method for text steganography is proposed. Embedding and extraction algorithms have been written and then coded in matlab programming language. The two algorithms here differ from those algorithms of my previous paper [5] by many points. Even though both papers make use of the non-printable Unicode characters in their hiding method but they differ by the language of the secret and cover text. In [5] the secret and cover text is of English language only. While in current algorithms secret and cover text can be from any alphabetical language with cursive writing system such as Arabic or Arabic mixed with English text. Also implementation certain text hiding rules in current work helps in generalizing the hiding techniques and allowing it to be used for documents from many other text writing systems.

USEAGE OF ZWNJ AND ZWJ UNICODE CHARACTERS IN DATA HIDING TECHNIQUES

Zero-width non-joiner (ZWNJ) and zero-width joiner (ZWJ) are non-printing Unicode characters used in the computerized writing system and typesetting of some complex scripts such as Arabic and Indic scripts.

When character ZWNJ is placed between two characters that would otherwise be connected into a ligature, a ZWNJ causes them to be printed in their final and initial forms, respectively. The ZWNJ is encoded in Unicode as U+200C or 8204_{10}.

The character ZWJ has an opposite effect to that of ZWNJ, i.e., when it is placed between two characters that would otherwise not be connected, a ZWJ causes them to be printed in their connected forms. The character's code point is U+200D or 8205_{10}.

Since both characters have zero widths (and thus cannot be seen or displayed digitally and neither not appeared when printed by any computer printer), thus such characters can be used in steganography algorithms for hiding text in another text data.

Two techniques are there to use ZWNJ and ZWJ characters in hiding text-in-text. In both techniques, ZWNJ and ZWJ characters will replace the binary bits 0 and 1, of the corresponding data binary code of the value to be hidden, respectively. In the first technique, every binary bit of the character code of the character to be hidden is replaced directly with either ZWNJ or ZWJ. For example the English letter ‘A’ has an ASCI number of $(65)_{10}$ or $(0100\ 0001)_2$and with use of ZWNJ and ZWJ representations, the ‘A’ binary code will be change to (200C 200D 200C 200C 200C 200C 200D). While in the second technique, every character in the language of the hidden string must be coded first according to some coding system but with the use of characters ZWNJ and ZWJ as their coding bits. For example, suppose the codes assigned to the English letters ‘A’,‘B’,and ‘C’ are ‘200C’ , ‘200D’, and ‘200C 200C’ respectively. Thus to hide ‘A’ for instance, one must use its corresponding code (i.e., ‘200C’) and also, to hide ‘C’ one can replace it with its new code (‘200C 200C), and so on. The second technique will be preferable when the required storage space or memory for the stego text (secret text within cover text), are taken into consideration.

In general sense, the above two techniques can be implemented in any text-in-text hiding algorithm; except in some situations,there is one important point must be taken into consideration. This point is related to the under used cover text language. Characters ZWNJ and ZWJ are originally added to the Unicode characters set for those languages

(including Arabic, Persian, Urdu, Pashto, Baloch, Malay, and some other languages), in which word is written in cursive style or by joining characters of each word one with its preceding or subsequent character. In such languages, each character may have more than one displayed forms (shapes or pictures), and also each character shape may change to another shape depending on its position or location within that word. In other words, it is not appropriate to use ZWNJ and ZWJ characters directly in hiding text into a cover text from such languages but must apply special insertion rules to the new codes before doing so, (some of these rules will be used in this paper). Also these two characters (ZWNJ and ZWJ), can be directly used (no need to look for the insertion rules), in text hiding in a cover text from other languages (that usually write their words in separated or non-connected character forms), like English and most of Europeans languages, and all other languages in which each character has one shape or form irrespective of its location in the word.

ARABIC EXTERNAL CODING SYSTEM

Text-in-text hiding technique by Unicode characters approach requires all text characters to be encoded in at least two different coding systems. **Firstone for the cover text**, it is internal to the computer system. It should be of the same general type of computer character coding system such as the ASCII or Unicode characters set. **The second code system is for the secret text**. It is an external coding system. It is preferred to be different from the first one. To minimize memory space required to store secret text, and thus enhancing processing speed, the second coding system must be constructed with minimum redundancy codes.

Huffman code generates a variable length code table for encoding a source symbol. It creates the code table depending on frequency of occurrence for each possible value of the source symbol [6]. In this work, Huffman coding is used to code Arabic secret text, according to Arabic frequency table derived from **intellaren.com** [7] with use of ZWNJ and ZWJ characters as bits of coding. Table (1) shows only small part of our used external coding system. The real used table has much more coding rows which are enough for encoding all remaining Arabic marks, punctuations, numbers and other symbols, also it contains the codes for most of English characters (Letters, numbers, punctuations, and other English symbols), to allow mixing between Arabic and English text.

Table 1: External Arabic Coding-Table

Rank	Letter	Code	Rank	Letter	Code
1	space	ZWNJ	20	ذ	ZWNJ ZWJ ZWNJ ZWJ
2	ا	ZWJ	21	ح	ZWNJ ZWJ ZWJ ZWNJ
3	ل	ZWNJ ZWNJ	22	ج	ZWNJ ZWJ ZWJ ZWJ
4	ن	ZWNJ ZWJ	23	ي	ZWJ ZWNJ ZWNJ ZWNJ
5	م	ZWJ ZWNJ	24	خ	ZWJ ZWNJ ZWNJ ZWJ
6	و	ZWJ ZWJ	25	خ	ZWJ ZWNJ ZWJ ZWNJ
7	ي	ZWNJ ZWNJ ZWNJ	25	ة	ZWJ ZWNJ ZWJ ZWJ
β	ه	ZWNJ ZWNJ ZWJ	26	ش	ZWJ ZWJ ZWNJ ZWNJ
9	ر	ZWNJ ZWJ ZWNJ	27	ص	ZWJ ZWJ ZWNJ ZWJ
10	ب	ZWNJ ZWJ ZWJ	28	ض	ZWJ ZWJ ZWJ ZWNJ
11	ت	ZWJ ZWNJ ZWNJ	29	ز	ZWJ ZWJ ZWJ ZWJ
12	ك	ZWJ ZWNJ ZWJ	30	ء	ZWNJ ZWNJ ZWNJ ZWNJ ZWNJ
13	ع	ZWJ ZWJ ZWNJ	31	أ	ZWNJ ZWNJ ZWNJ ZWNJ ZWJ
14	أ	ZWJ ZWJ ZWJ	32	ث	ZWNJ ZWNJ ZWNJ ZWJ ZWNJ
15	ف	ZWNJ ZWNJ ZWNJ ZWNJ	33	ظ	ZWNJ ZWNJ ZWNJ ZWJ ZWJ
16	ق	ZWNJ ZWNJ ZWNJ ZWJ	34	غ	ZWNJ ZWNJ ZWJ ZWNJ ZWNJ
17	س	ZWNJ ZWNJ ZWJ ZWNJ	35	ئ	ZWNJ ZWNJ ZWJ ZWNJ ZWJ
18	د	ZWNJ ZWNJ ZWJ ZWJ	36	ظ	ZWNJ ZWNJ ZWJ ZWJ ZWNJ
19	!	ZWNJ ZWJ ZWNJ ZWNJ	37	ؤ	ZWNJ ZWNJ ZWJ ZWJ ZWJ

Rules for Arabic Text Data Hiding

Arabic is one of the most widespread writing systems in the world (about 1 billion people 14% use the Arabic alphabet [8]). It is an alphabet script. Arabic is written and read from right to left. The Arabic alphabet comprises 28 letters. Each letter has a basic form. This form often changes depending on whether the letter is placed at the beginning, middle or end of the written word [Example: the Arabic letter, ع 'ayn' is written as عـ when it comes at the beginning and as ـعـ at the middle and ـع or ع at the end of the word]. In addition to the 28 Arabic basic letters and their different contextual forms, there are many other characters use in Arabic for different purposes such as marks placed above or below other letters or as punctuations, ornaments, word ligatures, and extended Arabic forms. Unicode contains approximately 1.100 code points for Arabic characters [9].

The idea of using non-printing Unicode characters in text-in-text hiding techniques becomes well known for hiding any secret text in English cover text. It can be done in straight forward way by replacing the secret text letter corresponding code by the non-printing characters (ZWNJ and ZWJ) code, and then inserting it before or after any selected letter of the cover text. This is not true and not the case for all world languages, such Arabic. That is due to the followings reasons:

- Arabic text letters may change their informs depending on where they placed in the word.
- Insertion of ZWNJ and ZWJ characters between letters of an Arabic word may result in different effects on the shape or the appearance of that word. As shown in Figures (1 and 2).
- Some Arabic letters (ء آ أ ؤ إ ا ة د ذ ر ز و ى),are not allowed to be connected (joined) to the next letter in word letters sequence. i.e., those letters must have a very short space at the end of their forms to separate them from the next letter and there is no any possibility to be truly joined with next one in the text.

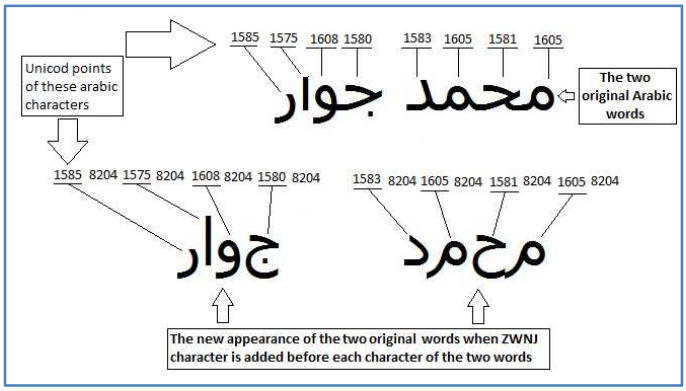

Figure 1: Effect of Inserting ZWNJ (Character Point 8204), in Front of Each Letter of the two Arabic words ' محمد جوار '

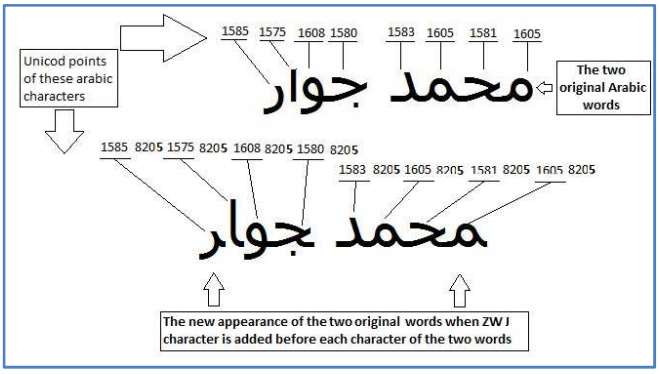

Figure 2: Effect of Inserting ZWJ (Character Point 8205), in Front of Each Letter of the Two Arabic Words ' محمد جوار '

- Two of the most frequently used Arabic letters 'ل' and ' ا ' when come in this order, and with no space separating them from each other, they will be given a special shape combining them together like ' لا ' and any separation, however it small, will change that shape to a different and unusual one ' لـا '.or ' ل ا '.
- All numbers, punctuation marks, mathematic symbols, and any characters from other language (other than Arabic), should not be joined from their both sides when hiding in Arabic text.
- Taking above reasons into consideration, one must derive rules to insert any character sequence of ZWNJ, ZWJ or both of them in between any Arabic word letters without changing its original picture or its original appearance. To do so, all Arabic characters must be divided first into four categories or groups**:**
- **First group** contains all Arabic characters which each have exactly four possible forms related to their place in the word (at the beginning, middle, end, and isolated forms).
- **Second group** contains all Arabic characters which have only two forms (at the middle and isolated forms).
- **Third group** is for those Arabic letters that often combined with each other to build another form.
- **Forth group** is to contain all remaining characters that have single and unique form (the isolated form).

Note: The fourth group also can contain any character from other languages.

To hide any secret code sequence of ZWNJ and ZWJ in Arabic cover text word, the following rules must be followed:

Note: Secret_code_sequence is a sequence of ZWNJ and ZWJ characters come out as result of running a coding system. These codes can only be used to replace characters from the secret text. While characters of the cover text have to be replaced only by real Unicode points obtained from the Unicode character table..

Rule – 1

Just before the beginning of the word of cover text or directly after the last character of it, the secret_code_sequence must be enclosed by two ZWNJ characters such that:

New secret_code_sequence = ZWNJ + secret_code_sequence + ZWNJ

Rule – 2

IF current and previous characters in current word of cover text, are both contained in third group **AND** meet the conditions of combination, **THEN** do not use this place to hide secret_code_sequence and skip to the next character of the cover text.

Rule - 3

IF previous and current characters of the current word of the cover text are both in fourth group **OR** just the previous character is contained in second group, **THEN** enclose the secret_code_sequenceby two ZWNJ characters such that:

New secret_code_sequence = ZWNJ + secret_code_sequence + ZWNJ

Rule-4

IF previous and current cover text characters are both contained in first group **THEN** enclose secret_code_sequence by two ZWJ characters such that:

New secret_code_sequence = ZWJ +secret_code_sequence + ZWJ

Figure (3) depicts the application of the hiding rules by using simple Arabic secret text **'م م م م' (**It doesn't mean any things, it is only repeating the Arabic later 'meem' four times), and using the Arabic word **'جوار'** (just a name), as cover text.

Note that the codes assigned to all character shown in Figure (3) are as in follow:

ZWNJ : $200C_{16} = 8204_{10}$ Obtained from the Unicode character set.

ZWJ : $200D_{16} = 8205_{10}=$

'ج' : $062C_{16} = 1580_{10}=$

'و' : $0648_{16}= 1604_{10}=$

'ا' : $0627_{16}= 1575_{10}=$

'ر' : $0631_{16}= 1585_{10}=$

'م' : $(200D\ 200C)_{16} = (8205\ 8204)_{10}$ the code assigned according to special

coding system use to encode secret text.

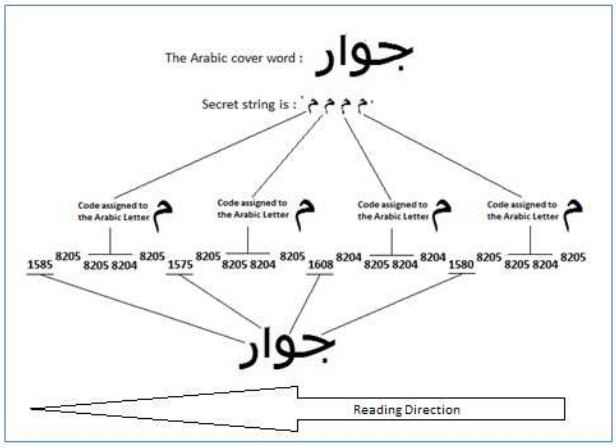

Figure 3: Simple Example on Implementation of the Hiding Rules

EMBEDDING ALGORITHM

- Start of embedding algorithm
- Read secret and cover text from input files.
- Format secret and cover text in separated lines of text.
- LOOP_WHILE still there secret or cover text lines are not processed do:

Assign text in next secret and cover lines to temporary variables of current_secret_line and current_cover_line respectively

Determine size of current_secret_line and current_cover_line

Apply the following two conditions:

Condition (1)

For each secret text line there must be a corresponding cover text line, and if not so, then append new line of spaces(spaces number must equal to size of current_secret_line), and call the new line as current_cover_line.

Condition (2)

Text size of current_cover_line must be greater or equal to size of current_secret_line, and if not then append number(equal to difference in size between the two current_cover_line and current_secret_line), of spaces to the end of the current_cover_line.

Apply the four Arabic text-in-text hiding rules to hide current_secret_line contents into current_cover_line contents to get stego text line.

Write stego text line into output file.

- End of LOOP_WHILE
- End of embedding algorithm.

EXTRACTION ALGORITHM

- Start of extracting algorithm.
- Read stego text from input file.
- Create the following two temporary variables for intermediate processing:

  Retrived_Code_Sequence,

  Retrived_Secret_Character and

  Retrived_Secret_String,
- LOOP_FOR each characters in stego text:

  IF current character is equal to either of ZWNJ or ZWJ character then

  Append it to Retrived_Code_Sequence.

  Go back to LOOP_FOR.

  ELSE

  Call function ‘tablesearch2.m’ to obtain the character corresponding to

  Retrived_Code_Sequencebythe used encoding system, and Assigned

  function returned-value to Retrived_Secret_Character.

  End_IF

  Append Retrived_Secret_Character to Retrived_Secret_String.
- End of LOOP_FOR.
- Output Retrived_Secret_String
- End of extracting algorithm.

Note: Embedding and Extracting algorithms and all other functions are written as Matlabm-files and are shown in APPENDIX-A.

EXPERIMENTS ON THE PROPOSED APPROACH

Embedding and extraction sample files [of only Arabic secret text sample file Figure (4), into only Arabic cover Text sample file Figure (5) and also Arabic mixed with English secret text Figure (7), into Arabic mixed with English cover text Figure (8)], have been done and the resultant stego text files of Figures (6 and 9) are obtained. Stego text files look exactly as the corresponding used cover text sample files. The only difference between cover text and the stego text that can be noticed by smart users, is that sometimes when number of secret text lines is greater than number of cover text lines, the stego text file, Figure (6), contains some extra empty lines appended at its end. And this is due to the conditions that must be met and verified before starting hiding any secret text line into cover text line as it was mentioned in the embedding algorithm.

سري وشخصي
الموضوع / تقييم بحث
تحية طيبة ...
لقد جرت العادة على تقييم البحوث العلمية المقدمة للمؤتمرات العلمية من قبل خبراء مختصين ونظرا لما نعهده فيكم من خبرة ومعرفة في موضوع اختصاصكم لذا نحيل اليكم البحث الموسوم "اخفاء النص الانكليزي بالصور الملونة" مع استمارة تقييم البحث ... آملين ان يصلنا ردكم خلال اسبوعين من تاريخه اعلاه مع الاوليات وفي حالة الاعتذار، تفضلكم بإعادة الاوليات خلال اسبوع.
مع التقدير ...

Figure 4: Arabic Secret Text Sample File

تحية طيبة ...
يسرنا اعلامكم بنية لجنة النشاطات والفعاليات الاجتماعية في رئاسة الجامعة عن قيامها بسفرة ترفيهية عائلية لمنتسبي الجامعة الى سواحل شط العرب بمنطقة السيبة جنوب مدينة البصرة.
مع التقدير ..

Figure 5: Arabic Cover Text Sample File

تحية طيبة ...
يسرنا اعلامكم بنية لجنة النشاطات والفعاليات الاجتماعية في رئاسة الجامعة عن قيامها بسفرة ترفيهية عائلية لمنتسبي الجامعة الى سواحل شط العرب بمنطقة السيبة جنوب مدينة البصرة.
مع التقدير ..

Figure 6**: Stego Text File (Arabic Secret in Arabic Cover Text**

النص السري Secret Textهو اي نص مكتوب باستخدام برنامج تحرير النصوص Notepad.exe الشغال في بيئةWindows .

Figure 7: Arabic and English Mixed Secret Text Sample File

لغات البرمجة Programing Languages هي لغات تستخدم اللغة الانكليزية English Language في كتابة خطوات خوارزمية Algorithm حل المشكلة على شكل إيعازات Statement واوامر Commands تتبع قواعد Grammar وسياق خاص بكل لغة منها.

Figure 8: Arabic and English Mixed Cover Text Sample File

لغات البرمجة Programing Languages هي لغات تستخدم اللغة الانكليزية English Language في كتابة خطوات خوارزمية Algorithm حل المشكلة على شكل إيعازات Statement واوامر Commands تتبع قواعد Grammar وسياق خاص بكل لغة منها.

Figure 9: Stego Text File of Arabic and English Mixed Secret and Cover Text Sample Files

CONCLUSIONS

- The proposed embedding and extracting algorithms are applicable not only to secret and cover of Arabic or Arabic mixed with English text but also can be used to hide secret in cover text from any alpabetic langauge

(Greek, Russian, Thai, Hebrew, …),into another text from other language as long as the alphabet of the other language are encoded into the external coding system of this method.

- In contrast to my previous paper [5], using hiding rulesin this workmakes the current technique more general and aplicable in hiding text from languages other than Arabic and English into another writing systems. .
- Size of stego text file is much more larger than the sum of only secret and cover text sizes. Many calculations show that size of stego text file is almost equal to the sum of cover text size and (3 to 5) times of the secret text size. This is due to the effects of implementing the hiding rules and the external code system to the secret text.

REFFERNCES

1. Salomon David., “Data Privacy and Security”, Publisher: Springer-Verlag, 2003.
2. R. Vill´an, S. Voloshynovskiy, O. Koval, J. Vila, E. Topak, F. Deguillaume, Y. Rytsar, and T. Pun, “Text Data-Hiding for Digital and Printed Documents: Theoretical and Practical Considerations”, Computer Vision and Multimedia Laboratory - University of Geneva, 2006.
3. Monika Agarwal,” Text Steganography Approaches: a Comparison”, International Journal of Network Security & Its Applications (IJNSA), Vol.5, No.1, January 2013.
4. M. S. Shahreza, and M. H. S. Shahreza, “Text steganography in SMS,” 2007 Int. Conf. on Convergence Information Technology, 2007, pp. 2260-2265.
5. M. J. Khami, “Unlimited Size of English Plain Text-in-Text Hiding Algorithm”, International Journal of Computer Science and Engineering (IJCSE), Vol-6 Issue-1, Dec-Jan 2017.
6. https://en.wikipedia.org/wiki/Huffman_coding. 2017.
7. http://www.intellaren.com/articles/en/a-study-of-arabic-letter-frequency-analysis. 2017.
8. http://www.worldstandards.eu/alphabets, “The world’s scripts and alphabets” , Last update: 15 January 2017.
9. https://www.key-shortcut.com/en/writing-systems/%EF%BA%95%EF%BA%8F%D8%A2-arabic-alphabet.

APPENDIX – A

ARABIC TEXT HIDING MATLAB M-FILE

1-Embedding Program

```
% Embeddingprog. foArabic text hiding by rules & Unicode characters method.
%
% Input: Secret and cover text files.
% Output: Stego text file.
%
clc; clear all;
%% Variables declaration.
```

```
[Letter, Letter_UniCode]=array_definition2 ();
ArabicLettersNotJoineToNextLetter=[1571,1572,1573,1575,1583,1584,1585,1586,1577,1569,1608,1632,1633,…
1638,1639,1640,1641,1634,1635,1636,1637,46,32];
Both Side Separation List= [10, 13, 32, 46];
%% Open secret and cover input text files.
WD=cd; % Current directory
% (1) Secret text file opening
[filen1 pth1] =uigetfile({'*.txt';},'Choose Secret Text File: ');
ifisequal(filen1,0) || is equal(pth1,0)
cd(WD);
return % User cancelled.
end
Secret File Name= [pth1 filen1];
fs = fopen(Secret File Name, 'r');
Sec=fread(fs,'*ubit16');
Sec=Sec';
% Separate secret text into lines
Sec Lines SB=strfind (Sec,[char(13),char(10)]);
ifisempty(Sec Lines SB)
Sec Lines SB= [size (Sec, 2) +1];
end
Sec Lines No=size (SecLinesSB, 2);
fori=1:SecLinesNo
ifi==1
Sec Line Content {i}=double(Sec(1:SecLinesSB(i)-1));
else
Sec Line Content {i} =double (Sec (Sec Lines SB (i-1) +1: SecLinesSB (i)-1));
end
end
% (2) Cover text file opening
```

```
[filen2, pth2] =uigetfile({'*.txt';},'Choose Cover Text File: ');
ifisequal(filen2,0) || isequal(pth2,0)
cd(WD);
return % User cancelled.
end
Cover File Name= [pth2 filen2];
fc = fopen(Cover File Name, 'r');
Cov=fread(fc,'*ubit16');
Cov=Cov';
% Separate cover text into lines
Cov Lines SB=strfind (Cov, [char (13), char (10)]);
ifisempty(Cov Lines SB)
Cov Lines SB=[size(Cov,2)+1]; % It is only one line text.
end
Cov Lines No=size(CovLinesSB,2);
fori=1:CovLinesNo
ifi==1
Cov Line Content{i}=double(Cov(1:CovLinesSB(i)-1));
else
Cov Line Content {i}=double(Cov(CovLinesSB(i-1)+1:CovLinesSB(i)-1));
end
end
%(3) Stego Filename (To store stegano Text).
Output File Name= [pth2,'M_', filen2];
Fm=fopen(Output File Name, 'w');
%
L Sec=0; L Cov=0;
%% Start hiding algorithm
while (L Sec<=Sec Lines No)|| (L Cov<=Cov Lines No)
```

```
L Sec=LSec+1; L Cov=LCov+1;
ts='';tc='';
if L Sec<=Sec Lines No
% Read one line from secret text.
ts=char(cell2mat(Sec Line Content(L Sec)));
end
if L Cov<=Cov Lines No
% Read one line from cover text.
tc=char(cell2mat(Cov Line Content(L Cov)));
end
sizets=size(ts,2); % Size of current secret text line .
sizetc=size(tc,2); % Size of current cover text line .
%
% Before starting hiding secret text lines in cover text lines the
% following conditions must be met: -
% 1) For each line of secret text there must be a corresponding cover
% text line, and if not so, then append new line of spaces to cover lines.
% 2) Each line of the cover text must contain number of characters at
% least equal to those of the corresponding secret text line
%
% The above two conditions are performed as in follow:
%
ifsizets>sizetc
ifsizetc==0
tc= [repmat(' ',1,sizets)]; % Condition (1)
else
tc= [tc(1: sizetc),repmat(' ',1,sizets-sizetc)]; % Condition (2)
end
end
```

```
sizets=size(ts,2); % Determine size of current secret text line .
sizetc=size(tc,2); % Determine size of current cover text line .
 %% Hiding Arabic line of secret text into Arabic line of cover text.
pc=1;ps=1; % Counters for current cover & secret characters in each line.
stegt=[]; % Empty Stego text line.
whilesizetc>=pc
if (ps>sizets)
stegt=[stegt,tc(pc)];pc=pc+1;continue;
end
Given String=ts(ps); % Selection of one character from secret text.
% Function 'tablesearch2.m' to get the new code for the
% selected secret character 'Give String' by applying different code
% system (code in the form of ZWNJ and ZWJ sequence)
[Cell Name, Cell Uni Code]=tablesearch2 (Given String, 1);
Cell Uni Code=char (Cell Uni Code); % Convert from Unicode point to char.
% Function 'Concat_Op.m' works according to our Arabic text-in-text
% hiding rules.
if pc==1
ch=Concat_Op(Cell UniCode,' spsn');
stegt=[char(ch),tc(pc)];
else
dctpc=double(tc(pc));dctpc1=double(tc(pc-1));
if (dctpc1==1604 &&dctpc==1575) || dctpc==10 || dctpc==13
stegt=[stegt, tc(pc)];
elseifismember(dctpc, BothSideSeparationList)|| ...
ismember(dctpc1,BothSideSeparationList)
ch=Concat_Op(Cell UniCode,' spsn');
stegt=[stegt, char(ch),tc(pc)];
elseif (ismember(dctpc1,ArabicLettersNotJoineToNextLetter))
```

```
ch=Concat_Op(CellUniCode, 'spsn');
stegt=[stegt, char(ch),tc(pc)];
else
ch=Concat_Op(Cell UniCode, 'jpjn');
stegt=[stegt, char(ch),tc(pc)];
end
end
pc=pc+1; ps=ps+1;
end
stegt=[stegt, char(13),char(10)];
fwrite(fm, stegt,'*ubit16');
end
%% End of hiding section
fclose all; % Close all opened files.
cd(WD); % Return to original working directory.
% End of embedding prog.
```

2-Extracting Program

```
% Extractingprog. of Arabic text hiding by rules and Unicode characters method.
% Input: Stego text file.
% Output: Secret text file.
%
clc; clear all;
% Read stego text filename.
[filen1 pth1] =uigetfile({'M_*.txt';},'Choose Stego Text Filename: ');
ifisequal(filen1,0) || isequal(pth1,0)
return % User cancelled.
end
Stego File Name= [pth1 filen1];
fs = fopen(Stego File Name, 'r');
```

```
stegt=fread(fs,'*ubit16');
fclose (fs);
stegt=stegt';
%% Extract Secret text.
S1= []; S2= []; S3= [];
fori=1:size(stegt,2)
ifismember (double(stegt(i)),[8204,8205])
S1= [S1, stegt (i)];
else
if size(S1,2)>=2
S1=S1 (2: end-1); %size (secret1)
for k=1:size(S1,2)
 S2= [S2,' ', dec2hex (S1 (k), 4)];
end
S2=S2 (2: end);
[Cell Name, Cell UniCode]=tablesearch2 (S2, 3);
S3=[S3,CellName];
S1=[]; S2=[];
end
end
end
S3=char (S3); S3=S3';
fprintf('\n Recovered Secret Text is:\n\n %s\n\n',S3);
fclose all;
% End of extracting program
```

3-Function array_deffinition

```
function [Letter, Letter_UniCode]=array_definition2
global Letter;
globalLetter_UniCode;
```

```
% Creation of Letter, Letter_BinCode, and Letter_Unicode

Letter=[{char(32)};{char(1575)};{char(1604)};{char(1606)};{char(1605)};{char(1608)};{char(1610)};{char(1607)};

{char(1585)};{char(1576)};{char(1578)};{char(1603)};{char(1593)};{char(1571)};{char(1601)};{char(1602)};{char(1587)};{char(1583)};{char(1573)};{char(1584)};{char(1581)};{char(1580)};{char(1609)};{char(1582)};{char(1577)};{char(1588)};{char(1589)};{char(1590)};{char(1586)};{char(1569)};{char(1570)};{char(1579)};{char(1591)};{char(1594)};{char(1574)};{char(1592)};{char(1572)};{char(1632)};{char(1633)};{char(1634)};{char(1635)};{char(1636)};
{char(1637)};{char(1638)};{char(1639)};{char(1640)};{char(1641)};{char(1642)};{char(1643)};{char(1644)};{char(1645)}; {char(1548)};{char(1549)};

{'!'};{""};{'#'};{'$'};{'%'};{'&'};{""};{'('};{')'};{'*'};{'+'};

{','};{'-'};{'.'};{'/'};{'0'};{'1'};{'2'};{'3'};{'4'};{'5'};{'6'};

{'7'};{'8'};{'9'};{':'};{';'};{'<'};{'='};{'>'};{'?'};{'@'};{'['};

{'\'};{']'};{'^'};{'_'};{'`'};{'{'};{'|'};{'}'};{'~'};{char(13)}; {char(10)};{char(9)};{char(11)};{char(12)};{char(14)};

{char(15)};{char(16)};{char(17)};{char(18)};{char(19)};{char(20)};{char(21)};{char(22)};{char(23)};{char(24)};{char(25)};{char(26)};{char(27)};{char(28)};{char(29)};{char(30)};{char(31)};{char(8)};{char(7)};{char(6)};{char(5)};{char(4)};{char(3)};{char(2)};{char(1)}];

Letter_BinCode=[

{'0'};{'1'};{'00'};{'10'};{'01'};{'11'};{'000'};{'001'};{'010'};{'011'};{'100'};{'101'};{'110'};{'111'};{'0000'};{'0001'};{'0010'};{'0011'};{'0100'};{'0101'};{'0110'};{'0111'};{'1000'};{'1001'};{'1010'};{'1011'};{'1100'};{'1101'};{'1110'};{'1111'};{'00000'};{'00001'};{'00010'};{'00011'};{'00100'};{'00101'};{'00110'};

{'00111'};{'01000'};{'01001'};{'01010'};{'01011'};{'01100'};{'01101'};
{'01110'};{'01111'};{'10000'};{'10001'};{'10010'};{'10011'};{'10100'};{'10101'};{'10110'};{'10111'};{'11000'};{'11001'};
{'11010'};{'11011'};{'11100'};{'11101'};{'11110'};{'11111'};{'000000'};{'000001'};{'000010'};{'000011'};{'000100'};{'000101'};{'000110'};{'000111'};{'001000'};{'001001'};{'001010'};{'001011'};{'001100'};{'001101'};{'001110'};{'001111'};{'010000'};{'010001'};{'010010'};{'010011'};{'010100'};{'010101'};{'010110'};{'010111'};{'011000'};{'011001'};{'011010'};{'011011'};{'011100'};{'011101'};{'011110'};{'011111'};{'100000'};{'100001'};{'100010'};{'100011'};{'100100'};{'100101'};{'100110'};{'100111'};{'101000'};{'101001'};{'101010'};{'101011'};{'101100'};{'101101'};{'101110'};{'101111'};{'110000'};{'110001'};{'110010'};{'110011'};{'110100'};{'110101'};{'110110'};{'110111'};{'111000'};{'111001'};{'111010'};
{'111011'};{'111100'};{'111101'};{'111110'};{'111111'}];

Letter_UniCode=[];

fori=1:size(Letter_BinCode,1)

te=char(Letter_BinCode(i,1));

dd=[];

for j=1:size(te,2)
```

```
ifte(1,j)=='0'
if j>1
dd=[dd,' ','200C'];
else
dd=['200C'];
end
else
if j>1
dd=[dd,' ','200D'];
else
dd=['200D'];
end
end
end
Letter_UniCode=[Letter_UniCode;{dd}];
end % End of array_deffinition2 function.
```

4-Function tablesearch2

```
function [Cell Name, Cell UniCode]=tablesearch2(Given String, field)
global Letter;
globalLetter_UniCode;
[Letter, Letter_UniCode]=Arabic_array_definition2;
%Note: field=1 if Letter is given
% field=2 if Letter_BinCode is given
% field=3 if Letter_UniCode is given
Cell Name={};Cell UniCode={};
ifnargin<2
return
end
if (field<1)||(field>3)|| isempty(Given String)
```

```
return
end
fori=1:size(Letter,1)
if field==1
id= strfind(Letter, Given String);
end
if field==2
id=strfind(Letter_Bin Code, Given String);
end
if field==3
id=strfind(Letter_UniCode, Given String);
end
if id{i,1}==1
break
end
end
if id{i,1}==1
CellName=Letter (i); CellUniCode=Letter_UniCode(i);
else
CellName={};Cell UniCode={};
end % End of tablesearch2 function.
```

5-Function Concat_Op

```
function Hidden_Code=Concat_Op(Hcode, concatination)
%{
separate=char(8204);joiner=char(8205);
spsn=[char(8204),char(8204)] ; jpjn=[char(8205),char(8205)];
spjn=[char(8204),char(8205)] ; jpsn=[char(8205),char(8204)];
%}
Hidden_Code=[];
```

```
ifnargin<2
return
end
ifisempty(Hcode)
return
end
s0=[];
fori=1:5:size(Hcode,2)
s1=Hcode(i:i+3);s2=hex2dec(s1);
ifi==1
s0= [s0, s2];
else
s0= [s0, s2];
end
end
switchconcatination
Case 'spsn'
Hidden_Code=[8204,s0,8204]; %spsn
Case 'jpjn'
Hidden_Code=[8205,s0,8205]; %jpjn
Case 'spjn'
Hidden_Code=[8204,s0,8205]; %spjn
Case 'jpsn'
Hidden_Code=[8205,s0,8204]; %jpsn
Otherwise
Hidden_Code=[];
End % End of Concat_Op Function.
```