**CHAPTER 1**

**INTRODUCTION**

**1.1 Objective**

Computer databases are an increasingly necessary tool for organizing the vast amounts of biological data currently available and for making it easier for researchers to locate relevant information. By 1983, there were more than 2,000 sequences stored in GenBank, with a total of one million base pairs. There has been an instant growth to contain over 95 billion base pairs, reflecting an exponential growth rate in which the amount of stored data has doubled every 18 months as shown in the Figure 1.1 below. This reflects a need for compression of DNA sequences. Our software is totally self contained and works relatively as efficient as other packages related to the subject. It provides simple database rather than complex ones for high requirements and it provides good and easy graphical user interface to new, naive as well as experienced users of the computers. The information about the DNA sequences are stored in molecular biology databases. A DNA sequence file occupies more space in database. The compression algorithms that were used in the existing systems were not effective. In order to compress the file size effectively we are using DNA Sequence Compression Algorithm.

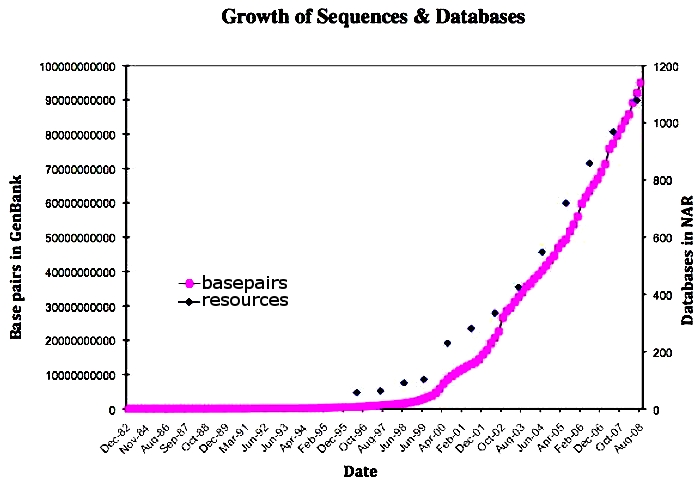


Figure 1.1 Graph showing the growth of sequences and databases

**1.2 DNA Structure**

DNA, or deoxyribonucleic acid, is the hereditary material in humans and almost all other organisms. Nearly every cell in a person’s body has the same DNA. Most DNA is located in the cell nucleus where it is called nuclear DNA, but a small amount of DNA can also be found in the mitochondria where it is called [mitochondrial DNA](http://ghr.nlm.nih.gov/chromosome/MT).

The information in DNA is stored as a code made up of four chemical bases: adenine (A), guanine (G), cytosine (C), and thymine (T) as shown in the Figure 1.2. Human DNA consists of about 3 billion bases, and more than 99 percent of those bases are the same in all people. The order, or sequence, of these bases determines the information available for building and maintaining an organism, similar to the way in which letters of the alphabet appear in a certain order to form words and sentences.

DNA bases pair up with each other, A with T and C with G, to form units called base pairs. Each base is also attached to a sugar molecule and a phosphate molecule. Together, a base, sugar, and phosphate are called a nucleotide. Nucleotides are arranged in two long strands that form a spiral called a double helix. An important property of DNA is that it can replicate, or make copies of itself.

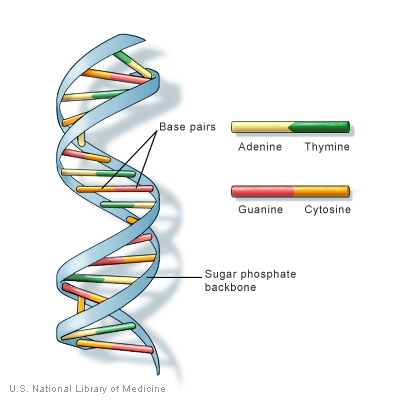


Figure 1.2 DNA Structure

**1.3 DNA Sequence**

DNA sequencing includes several methods and technologies that are used for determining the order of the [nucleotide](http://en.wikipedia.org/wiki/Nucleotide) bases. The DNA sequence has capacity to represent [information](http://en.wikipedia.org/wiki/Information). Biological DNA represents the information which directs the functions of a living thing. In that context, the term genetic sequence is often used. Sequences can be read from the biological raw material through [DNA sequencing](http://en.wikipedia.org/wiki/DNA_sequencing) methods. Normally, every person carries two variations of every [gene](http://en.wikipedia.org/wiki/Gene), one inherited from their mother, the other inherited from their father. The [human genome](http://en.wikipedia.org/wiki/Human_genome) is believed to contain around 20,000 - 25,000 genes. DNA sequencing is the process of determining the [nucleotide](http://en.wikipedia.org/wiki/Nucleotide) sequence of a given [DNA](http://en.wikipedia.org/wiki/DNA) fragment.

The sequence of the DNA of a living thing encodes the necessary information for that living thing to survive and reproduce. Therefore, determining the sequence is useful in fundamental research into why and how organisms live, as well as in applied subjects. Because of the importance of DNA to living things, knowledge of a DNA sequence may be useful in practically any biological [research](http://en.wikipedia.org/wiki/Research). For example, in [medicine](http://en.wikipedia.org/wiki/Medicine) it can be used to identify, [diagnose](http://en.wikipedia.org/wiki/Diagnosis) and potentially develop [treatments](http://en.wikipedia.org/w/index.php?title=Treatment_(medicine)&action=edit&redlink=1) for [genetic diseases](http://en.wikipedia.org/wiki/Genetic_disease). Similarly, research into [pathogens](http://en.wikipedia.org/wiki/Pathogens) may lead to treatments for contagious diseases. [Biotechnology](http://en.wikipedia.org/wiki/Biotechnology) is a burgeoning discipline, with the potential for many useful products and services. The DNA in an organism's [genome](http://en.wikipedia.org/wiki/Genome) can be analyzed to [diagnose](http://en.wikipedia.org/wiki/Medical_diagnosis) vulnerabilities to inherited [diseases](http://en.wikipedia.org/wiki/Disease). In addition to studying [chromosomes](http://en.wikipedia.org/wiki/Chromosome) to the level of individual genes, genetic testing in a broader sense includes [biochemical](http://en.wikipedia.org/wiki/Biochemical) tests for the possible presence of [genetic diseases](http://en.wikipedia.org/wiki/Genetic_disease), or mutant forms of genes associated with increased risk of developing genetic disorders.

Genetic testing identifies changes in chromosomes, genes, or proteins.[[5]](http://en.wikipedia.org/wiki/Nucleic_acid_sequence#cite_note-4) Usually, testing is used to find changes that are associated with inherited disorders. The results of a genetic test can confirm or rule out a suspected genetic condition or help determine a person's chance of developing or passing on a genetic disorder. Several hundred genetic tests are currently in use, and more are being developed

A sequence alignment is a way of arranging the sequences of [DNA](http://en.wikipedia.org/wiki/DNA), [RNA](http://en.wikipedia.org/wiki/RNA), or [protein](http://en.wikipedia.org/wiki/Protein) to identify regions of similarity that may be a consequence of functional, [structural](http://en.wikipedia.org/wiki/Structural_biology), or [evolutionary](http://en.wikipedia.org/wiki/Evolution) relationships between the sequences.

Genetic disorders are caused by the mutations or abnormalities that occur in a chromosome or genome. These abnormalities may appear phenotypically at any time of a human life. It is being estimated that there are around 4000 genetic disorders which affect the human life.

**DNA Sequencing Technology**

**BLAST:**

BLAST is the Basic Local Alignment Search Tool.  BLAST is a computer algorithm that is available for use online at the [National Center for Biotechnology Information (NCBI) website](http://blast.ncbi.nlm.nih.gov/Blast.cgi) and many other sites. BLAST can rapidly align and compare a query DNA sequence with a database of sequences, making it a critical tool to ongoing genomic research. BLAST is one of the most widely used bioinformatics programs, because it addresses a fundamental problem and the algorithm emphasizes speed over sensitivity. BLAST is more time efficient by searching only for the more significant patterns in the sequences, but it is comparatively sensitive.

**CHAPTER 2**

**LITERATURE REVIEW**

**2.1 Compression Algorithms**

**2.1.1 Understanding the properties of DNA sequences using Compression Algorithm[10]**

The information of DNA sequences, RNA sequences, and amino-acid sequences of proteins are stored in molecular biology databases. It is well known that the sizes of these databases are increasing nowadays very fast. Therefore it is needed to store and communicate data efficiently. DNA sequences contain four symbols ‘a,’ ‘t,’ ‘g,’ and ‘c,’ if these were totally random, the most efficient way to represent them would be using two bits for each symbol. However, only a small fraction of DNA sequences result in a viable organism, therefore the sequences which appear in a living organism are expected to be nonrandom and have some constraints. They should be compressible.

**2.1.2 Using DNA Sequences as a Challenging Subject for Compression Algorithms[10]**

DNA sequences only contain four symbols, therefore two bits per symbol is enough to represent these sequences even if they are totally random. However if one applies the standard text compression software such as compress or gzip, they cannot compress DNA sequences but only expand the file with more than two bits per symbol. Thus DNA sequences are important as a new challenge for study of compression algorithms. Huffman’s code also fails badly on DNA sequences both in the static and adaptive model, because there are only four kind symbols in DNA sequences and the probabilities of occurrence of the symbols are not very different. Concerning compression ratio, PPM is one of the best compression algorithms. However it cannot compress DNA sequences less than two bits per symbol either.

**2.1.3 Biocompress Algorithm [1]**

The algorithm Biocompress-2[1] is designed for the compression of DNA sequences. The classical text compression algorithms will not provide useful for DNA sequence compression.

In this algorithm the compression is done making use of the specific redundancies in DNA sequence like palindromes. Such repetitions do not exist in texts in natural or programming languages. Biocompress-2[1] detects such regularities and encodes them. To detect all factors independently o their size or position, it uses a window of arbitrarily large width, from first position to current position. At each step of the compression, the longest factor beginning at the current position and matching a factor which starts in the part of the chain already encoded is chosen. Biocompress detects an exact repeat in DNA using an automaton, and uses Fibonacci coding to encode the length and position of its previous location. If a subsequence is not a repeat, it is encoded by the naive 2 bits per symbol technique. The improved version, BioCompress-2 uses a Markov model of order 2 to encode non-repeat regions.

**2.1.4 Gencompress[12]**

Gencompress is based on approximate matching. The algorithm provides significantly better results using standard benchmark DNA sequences in both cases. The lossless compression algorithm Gencompress in this paper achieves significantly higher compression ratios than both Biocompress-2 and Cfact. Gencompress is able to detect approximate matches of any edit distance, including exact repeats. Significantly better compression results show that the approximate repeats are one of the main hidden regularities in DNA sequences. From Gencompress results a shared information distance value is approximately calculated which measures the relatedness between each pair of DNA sequences.

Table 2.1 Comparison of compression algorithms

|  |  |  |
| --- | --- | --- |
| Sequence | Bio - compress | Gen – compress |
| MTPACGA | 6.37% | 6.88% |
| MPOMTCG | 1.72% | 4.71% |
| HUMDYSTROP | 3.31% | 19.27% |
| HUMHBB | 8.17% | 16.35% |
| HUMGHCSA | 3.11% | 44.76% |
| HUMHPRTB | 4.08% | 9.04% |
| HUMHDABCD | 2.87% | 9.04% |

* 1. **Disease Identification**

DNA is more personal than a fingerprint. DNA sequences are increasingly becoming a part of the patient medical record. The effect of the discovery of DNA on scientific and medical progress has been enormous, which involves the identification of genes that trigger major diseases or the creation and manufacture of drugs to treat these devastating diseases. One important area of DNA research is that of genetics and medical research. Due to our discovery of DNA, our ability to actually diagnose diseases early on has been vastly improved. In addition, we have been able to better assess a person's genetic susceptibility to specific diseases. In doing so, we have also paved the pathway to formulate brand new drugs to treat these diseases.

DNA has become a prominent feature for disease identification as the cost of sequencing has been declining for over a decade due to automated sequencing while the storage capacity of computers has grown tremendously yet declined in price. Also many diseases are increasingly being found to have a DNA component, which can be used for diagnostic confirmation of the presence or absence of a disease. It acts as a probabilistic component that helps to establish the chances of being afflicted with a certain disease. For the purpose of disease identification using DNA sequences, the links [7][8] were referred.

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 Existing System**

**Gencompress**

Gencompress is based on approximate matching that gives good compression results on standard benchmark DNA sequence. In this algorithm the relatedness between two sequences or two genomes is measured. The lossless compression algorithm Gencompress achieves significantly higher compression ratios than both Biocompress-2 and cfact.

**Algorithm Steps**

**Step 1** For input *w*, assume that a part of it, say ‘v’, has already been compressed, and consider the remaining part as ‘u’ such that *w* = *vu*.

**Step 2** An optimal prefix in string u is selected and compared such that it approximately matches with some substring in ‘v’.

**Step 3**  If any match is found then that prefix part of *u* is encoded and the suffix of u is considered as w.

**Step 4**  This process continues till ‘u’ becomes null.

**Drawback**

Gencompress gives better compression ratio than the other algorithms; however when there are not enough approximate repeats it fails to achieve higher compression ratio.

**3.2 Proposed System**

**Sequence Compression Algorithm**

In proposed algorithm, **DNA-SCA** is able to detect more regularity and achieve best compression results. Apart from that, combinations of the DNA sequence are found which helps in easy analysis of characteristics of genome. Colouring of predefined combinations is helpful to differentiate the presence of combinations between the varieties of genes. All these applications are grouped together in one application tool.

**Operations**

The operations required by the users are:

* Data Compression
* Replacing Character Wise
* Identifying disease
* Effective Data Mining

**Algorithm Steps**

**Step 1** Initially the blank spaces in the sequence are identified and removed.

**Step 2**  Consider the input sequence as x. In x, the four character combinations like aaaa, cccc, tttt, gggg, actg, atcg, tcag etc., are selected and then those characters are encoded. The character combinations are found using the formula,

nPr = n! / (n-r)!

where,

n, r are non negative integers

r is the size of each permutation.

! is the factorial operator.

**Step 3** After encoding all the four character combinations, in the remaining part of x, the three character combinations are selected and are similarly encoded.

**Step 4**  Then finally the remaining two character combinations are encoded.

**Step 5**  This process continues till the character becomes singleton.

**Advantages**

* In the proposed system, the compression DNA Sequence Algorithm was used to compress the File.
* Sequence : ACTG
* DNA-SCA : 60% (Compression)
* Data tampering is reduced.
* Provides security that protect against outsider crime.
* Reduces cost of operations

**3.3 Feasibility Study**

Feasibility study is a test of the system proposal according to the workability, impact of the organization, ability to meet user’s needs and effective use of resources. The feasibility study must satisfy the following factors:

* User demonstrable needs
* Problem worth solving
* Method of solving problem.
  + 1. **Economical Feasibility**

Labs using DNA sequences are more concerned with accuracy than cost in future purchases. A significant number of respondents chose accuracy over cost, ease-of-use or speed[9]. The project provides frequently used methods for evaluating the effectiveness of a user system. It also provides savings and benefits to the users. Our project’s benefit does not outweigh the costs. The justification or alterations in the proposed system are made to have been approved. It is an on-going effort that improves the accuracy at each phase of the system life cycle.

* The automated system is not costly.
* Maintenance does not involve any investment in terms of money

Once the computerized system is installed, it can cater to the needs of the user, which is more cost-effective for the management. Therefore, the automated system is economically feasible.

* + 1. **Technical Feasibility**

Technical feasibility centers on the existing system. The existing system involves financial considerations to accommodate technical enhancements. The budget is not a serious constraint in our proposed project which is judged to be feasible. The users have understood the advantages, benefits and economic feasibility of the new system which is ready to afford the extra expense that may arise for the satisfaction of all the hardware and software requirements.

* + 1. **Operational Feasibility**

Users highly resist changing the computers to facilitate the changes. Our project installation provides a lot including the DNA sequence storage which is portable as it is stored in the database and also provides disease identification which is accessible by the users. The main solutions are

* Graphical display of compression results for easy readability.
* Visual GUI interface.
* The system is uniformly accepted by all the type of users.
* Abstraction of DBMS and connectivity from users.
  + 1. **Behavioral Feasibility**

The developed software behaved properly in different scenarios when deployed. It adapted itself and provided the users with appropriate outputs expected by them.

**CHAPTER 4**

**SYSTEM SPECIFICATION**

**4.1 Hardware Requirements**

System : Pentium IV 2.4 GHz

Hard Disk : 40 GB

Monitor : 15 VGA colour

Mouse : Logitech

RAM : 256 MB

Keyboard : 110 keys enhanced

**4.2 Software Requirements**

Operating system : Windows XP Professional

Front End : Microsoft Visual Studio .Net 2008

Coding Language : VB.NET

Back End : SQL SERVER 2000

**4.3 Platform Selection**

**VB .NET**

VB .NET is a major component of Microsoft Visual Studio .NET suite. Net is a Framework in which Windows applications may be developed and run. The .NET version of Visual Basic is a new improved version with more features and additions. After these new additions, VB qualifies to become a full object-oriented language such as C++.

**Features of VB.NET**

* Object Oriented Programming language.
* Support of inheritance, overloading, interfaces, shared members and constructors.
* Supports all CLS features such as accessing and working with .NET classes, interaction with other .NET languages, Meta data support, common data types, and delegates.
* Structured exception handling.

**Enterprise Manager**

The enterprise manager is used in the project because it provided new features in designing our project, accessing the data and creating and managing the components. It provided links to more information on design tools to measure the efficiency of the existing components. It helped to link the topics on data access enhancements. With the help of visual studio enterprise edition the component performance and management of tools are enhanced.

**SQL Server 2000**

Microsoft SQL Server 2000 extends the performance, reliability, quality, and ease-of-use. Microsoft SQL Server 2000 includes several new features that make it an excellent database platform for large-scale online transactional processing (OLTP), data warehousing, and e-commerce applications. It also provides support for better database design which served helpful for our project.

**CHAPTER 5**

**SOFTWARE DESCRIPTION**

**5.1 Features of the Software**

In the project the sequence compression algorithm is used. It uses enterprise manager as its front end and SQL server as its back end.

Table 5.1 Comparison of Gencompress and SCA tools

|  |  |  |
| --- | --- | --- |
| **FEATURES** | **GENCOMPRESS TOOL** | **SCA TOOL** |
| Algorithm used | Gencompress | Sequence compression |
| Compression rate | Only 40% | Upto 60% |
| Database | It is not possible to store in the contents in the database. | It is possible to store the contents in the database. |
| Disease Identification | No such options. | The sequence along with the virus name is identified. |

**5.2 Menu Options**

Initially the user or the admin have to login using valid username and password. Then the menu screen is opened only with the disease identification module for the user login and with all the options such as Genpack content module, Compression module, Disease identification module, Graphical comparison for the admin login.

When the user login into the account, he selects the disease identification module. As soon as it is clicked, a screen opens with the option for entering the patient id which when selected the DNA sequence of the given id is loaded. Then by clicking ‘identify disease’ option, the DNA sequence containing the disease is displayed along with the virus name causing the disease.

When it’s an admin, select the Genpack content module. In this a patient’s profile can be created, deleted, searched, saved and cleared if necessary. Then the admin selects the DNA compression module in which the DNA content for the given patient id is loaded. As soon as it is loaded apply encode button is selected and then apply DNA compress button is clicked. Now compression starts and in the DNA sequence four character, three character and two character datas are compressed as a dot as per the Sequence Compression Algorithm. The file size before compression and after compression is displayed below along with the reduced percentage after compression.

After compression, the admin selects disease identification module. In this again the patient id is selected in the combo box. The DNA sequence for the given id will be loaded. Then identify disease button is selected. Now the sequence containing the disease along with the virus name is displayed.

The compression ratio can be checked and easily identified using the graphical comparison chart in which the patient id is given in the x-axis and the reduced percentage is given in the y-axis. The reduced percentage of the existing system and the proposed algorithm is compared and clearly noticed.

At any time the user or the admin can exit the account by selecting the exit button.

**CHAPTER 6**

**PROJECT DESCRIPTION**

**6.1 Problem Definition**

A DNA sequence file occupies more space in the database. There has been an increased growth from 2000 base pairs to 95 billion base pairs, reflecting an exponential growth rate in which the amount of stored data has doubled every 18 months. This reflects a need for compression of DNA sequences. The standard compression algorithms such as gzip or Biocompress cannot compress the DNA sequences, but only expand their size. Even though the Gencompress algorithm gives better compression ratio than the other algorithms; however when there are no enough approximate repeats it fails to achieve higher compression ratio. In order to achieve a better compression ratio and to reduce the file size a new algorithm called SCA has been proposed.

**6.2 Module Description**

**6.2.1 Login Module**

In this module the valid username and password are entered. If the admin login is given it goes to the module containing Genpack content module, disease module and graph comparison. If the user login is given it goes to the page containing only the disease module. It displays an error message when invalid password or username is entered.

**6.2.2 Genpack Content Module**

In this module the ACTG sequence for the corresponding patient id is loaded by clicking on the load content tab. This module also have additional features to create a new patient id, to delete an existing patient’s data, to save the content loaded, to search a dataset for the given patient id and also to clear the contents when not required.

**6.2.3 Gencompress Module**

This module uses the Gencompress algorithm which is an existing algorithm. In this module the patient’s id is loaded initially. Then the Gencompress algorithm is applied by selecting apply Gencompress button. This module also displays the file size before compression and also the size after compression.

**6.2.4 DNA Compression Module**

In this module initially the patient’s id who’s DNA sequence need to be compressed is loaded and the file size is also displayed. As soon as the sequence gets loaded, the compression starts by selecting the DNA compress button. After the compression gets finished, the compressed file size and the reduced percentage are displayed in this module.

**6.2.5 Disease Identification Module**

In this module, initially the patient’s id is entered and the ACTG sequence for the given id is loaded. Then the identify disease button is selected which searches the DNA sequence set for the disease sequence. If the disease is found it displays the DNA sequence with disease along with the disease causing virus name.

**6.3 Data Flow Diagram**

**Admin Login**

Find the matching sequence

Admin

Login

Database

Login

Obtaining details for comparison

Figure 6.1 (a) General Depiction of Admin login

**User Login**

Verify

User login

Login

Find the matching sequence

Figure 6.1 (b) General Depiction of User login

**6.4 Usecase Diagram**



Figure 6.2 Use case diagram

**6.5 Database Design**

**6.5.1 Login**

Table 6.1 Content of login table

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Null** |
| Username | Varchar2(20) | No |
| Password | Varchar2(20) | No |

**6.5.2 Disease**

Table 6.2 Content of disease table

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Null** |
| Disease\_name | Varchar2(30) | No |
| Sequence | Varchar2(50) | No |

**6.6 Input Design**

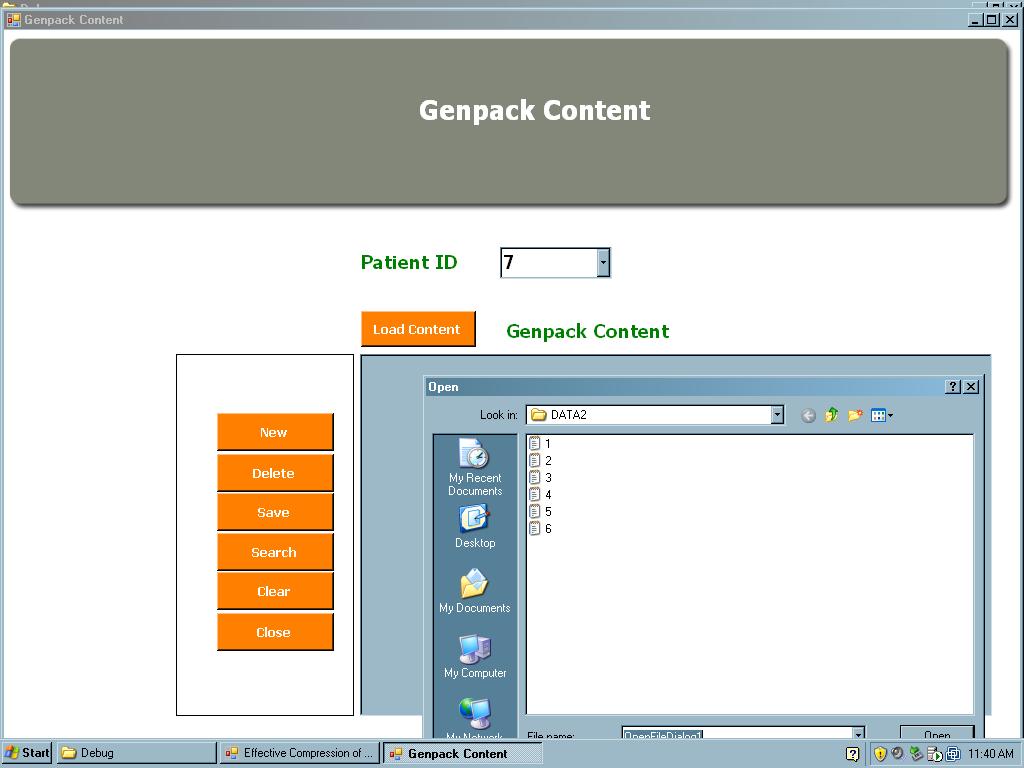
Input design is the most important part of the overall system design, which requires very careful attention. Often the collection of input data is the most expensive part of the system

Figure 6.3 (a) Loading content for the given id

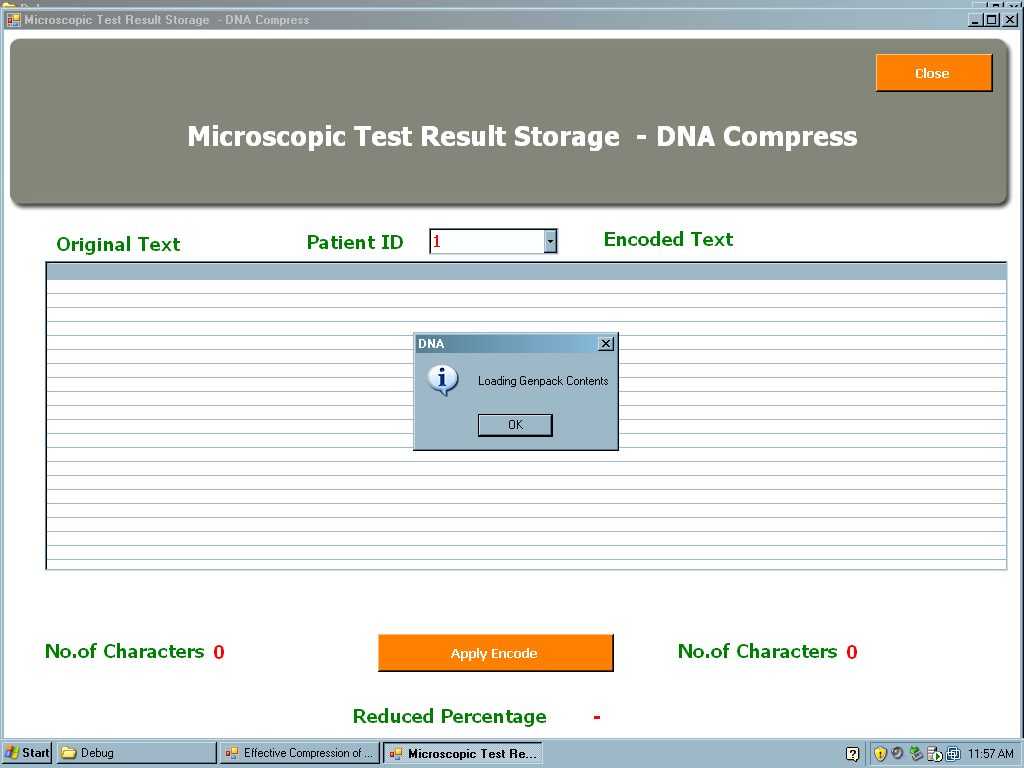
****

Figure 6.3 (b) Loading the contents for compression

As soon as the patient id is entered the DNA sequences are loaded and process completed message is seen.

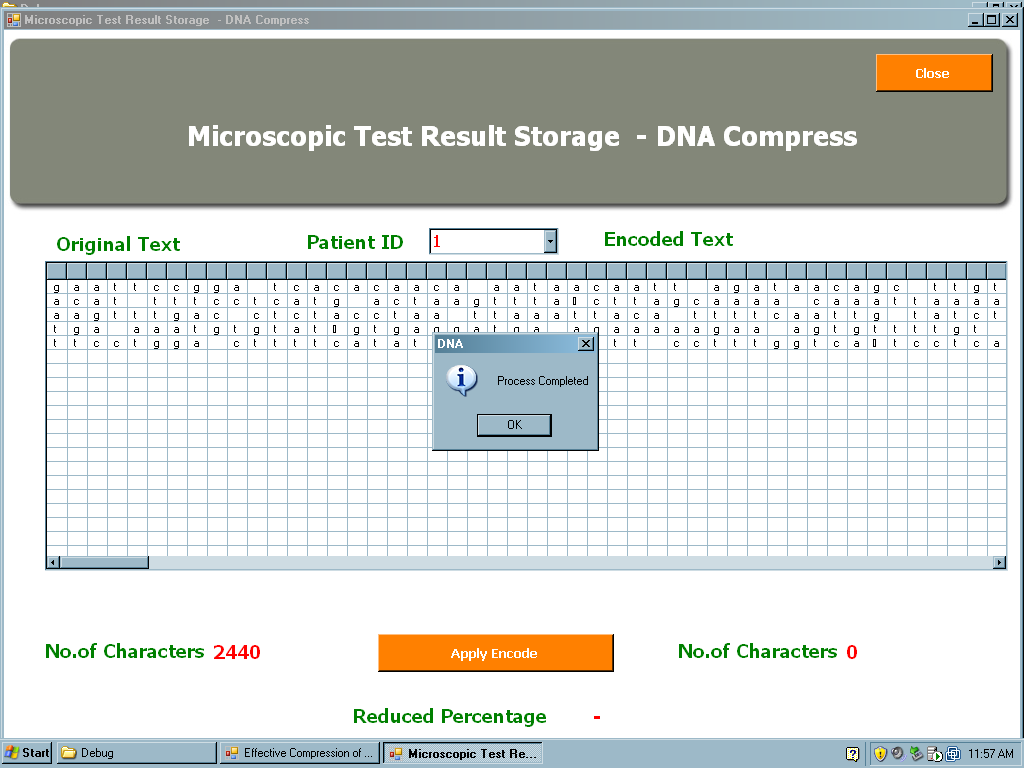
****

Figure 6.3 (c) After loading the contents

The content of DNA sequences are displayed for disease identification.

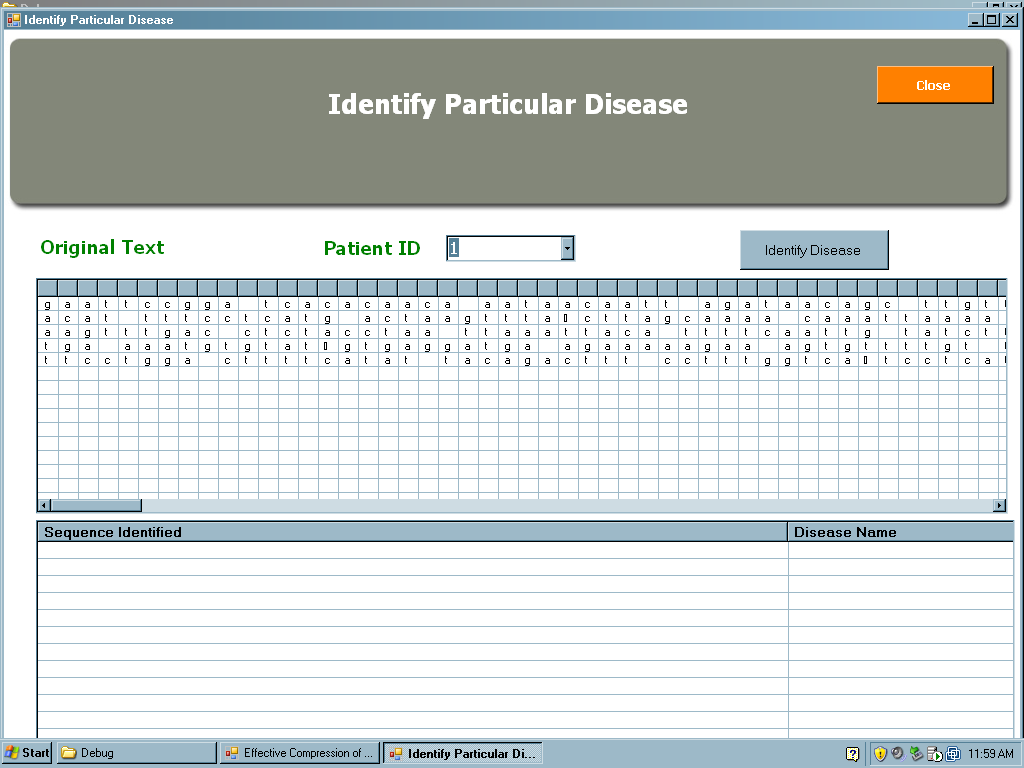
****

Figure 6.3 (d) Loading the contents for disease identification

**6.7 Output Design**

Computer output is most important and direct source of information to the user. Output from the computer system is required to communicate the result of processing to the user and to provide permanent copy of these results for later consultation.

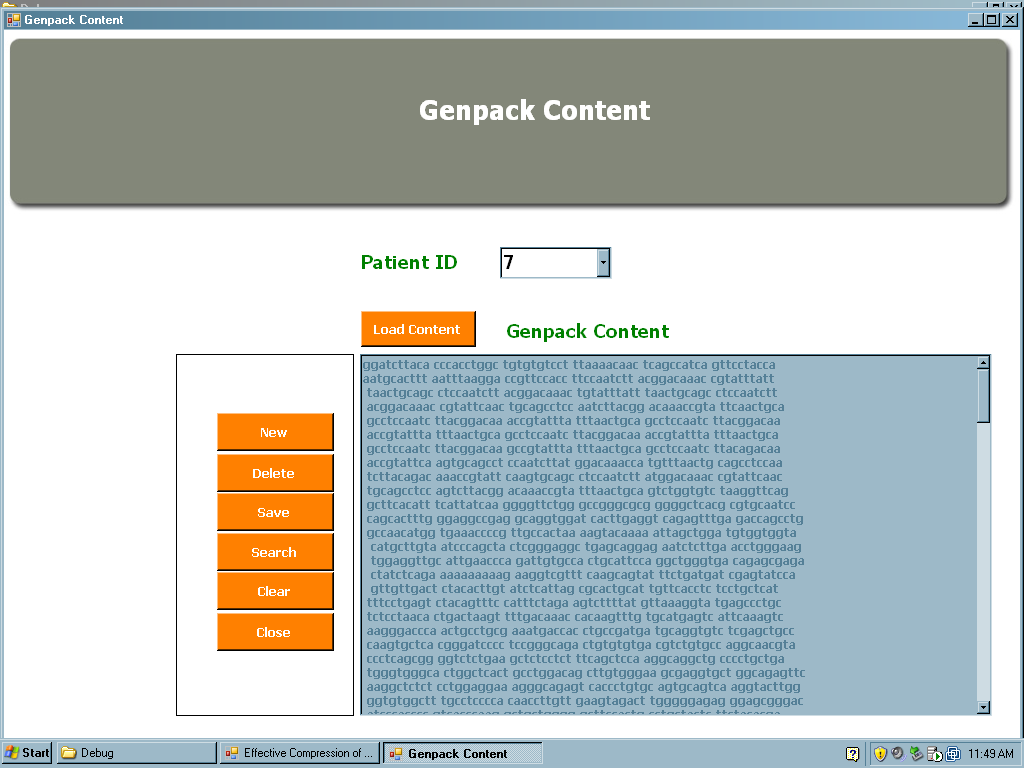


Figure 6.4 (a) Output screen after the contents are loaded

After entering the patient id and on selecting the compress option, the DNA sequences loaded are compressed and the file size is displayed.

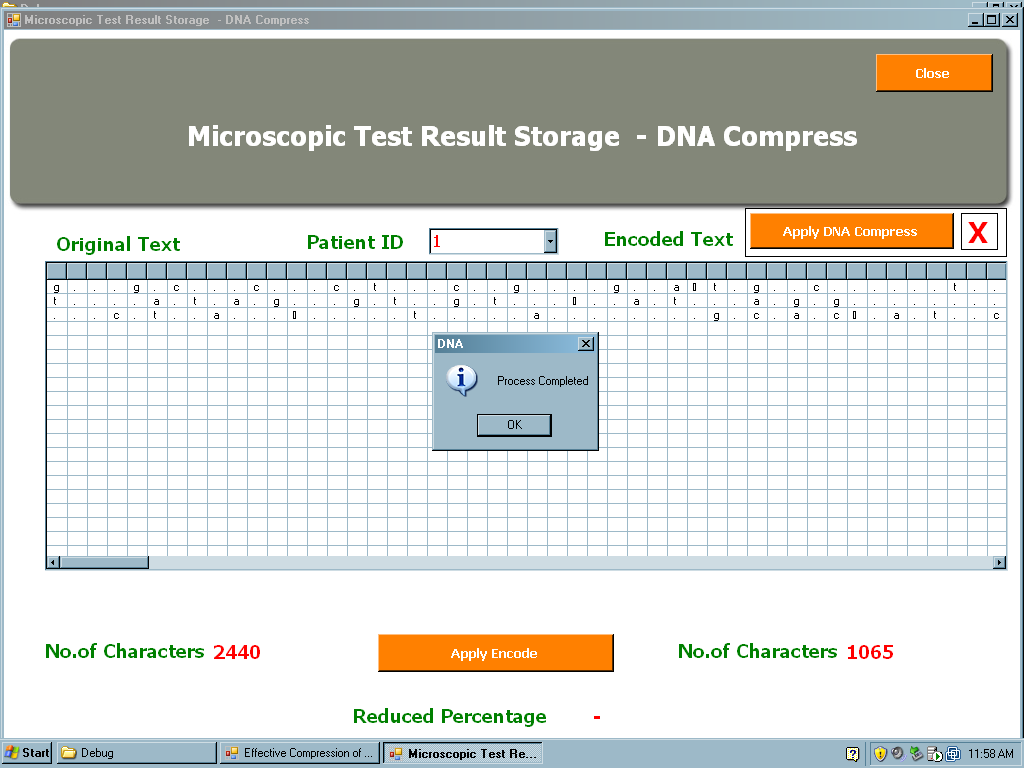


Figure 6.4 (b) Output screen after applying compression

For identifying disease the DNA contents are loaded and on selecting the identify disease option, the DNA sequence along with the disease name is displayed, else the error message is displayed.

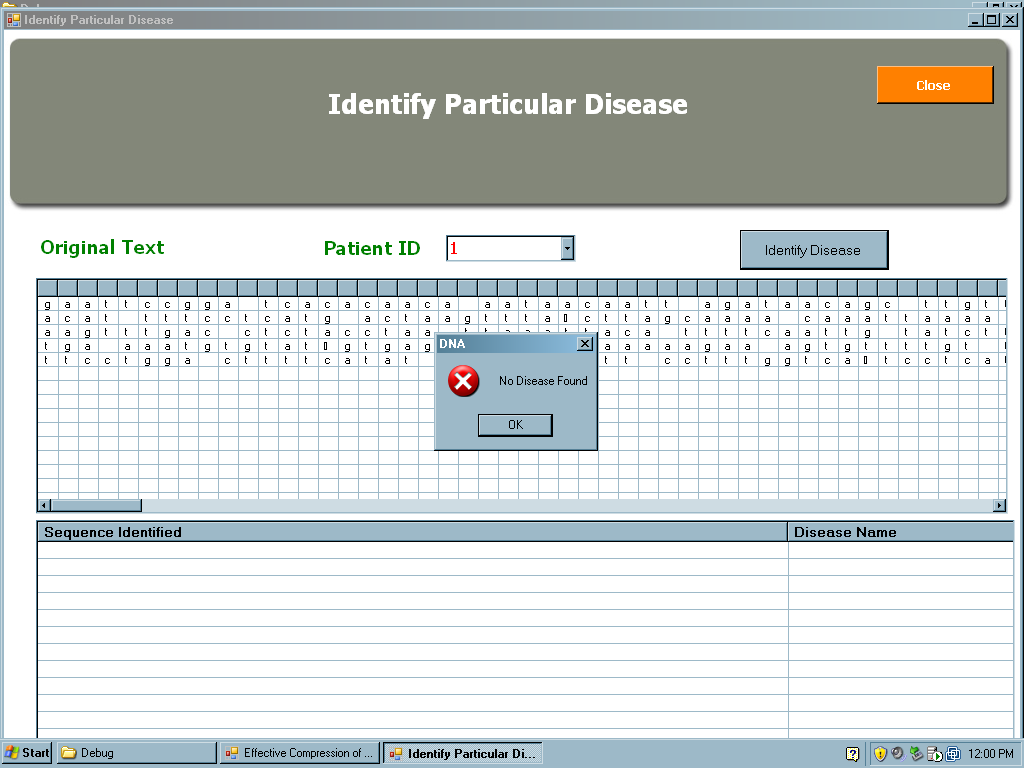


Figure 6.4 (c) Output screen for disease not found

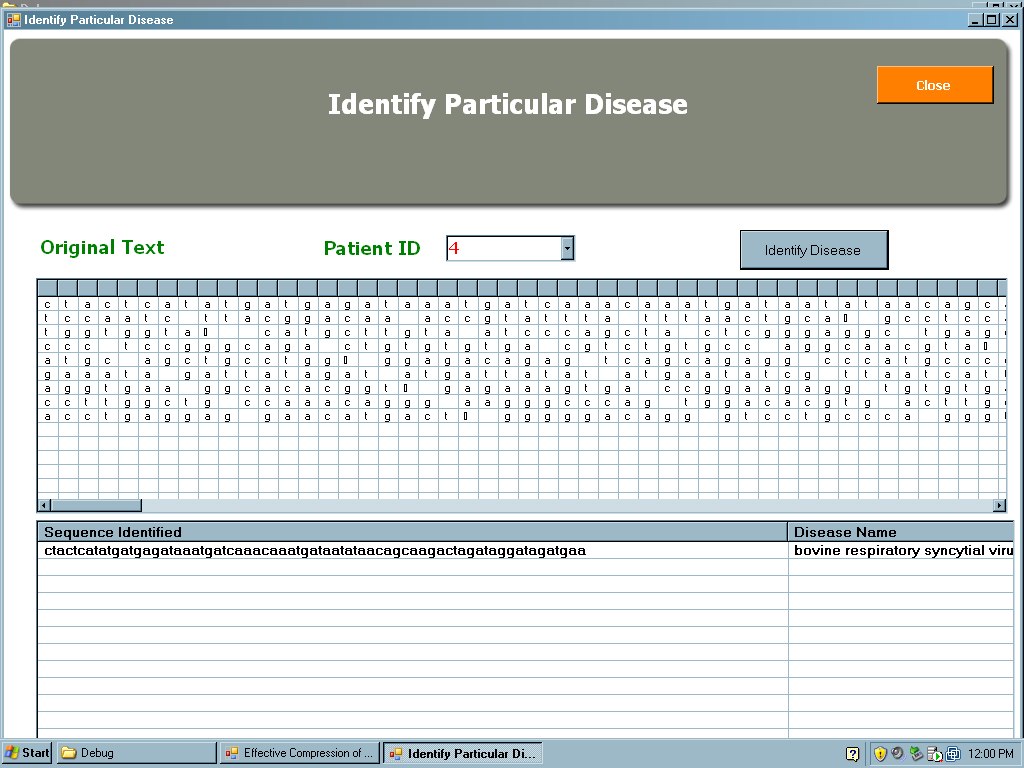


Figure 6.4 (d) Output screen displaying the disease

**6.8 Lookup Table Design**

The lookup table is designed with the information about the DNA sequences for the various viruses collected from various sources.

Table 6.3 Table displaying the disease and its sequence

|  |  |
| --- | --- |
| **DISEASE** | **SEQUENCE** |
| Human Herpesvirus 4 | ggcgggtgctggacgatgggcagcagatccgcgtgttctcctgcctcctggcggccgcccaaaaggaaaa |
|  | ttttccttttgggcggccgccaggaggcaggagaacacgcggatctgctgcccatcgtccagcacccgcc |
| Human Herpesvirus 8 | Acaggaaattcacaagcacctgttctgcgacccgctgtgcgccctcaatgctaaggtggt |
|  | Atggactggctctggtctacaaacggcgccctggcacgagccctgtccagtagcttaaat |
| Human Herpesvirus 5 | Tcagagttctttcccggagagttttcagccctcgctacgcctctatgtcgaaaatgtggctttattcatc |
|  | ggcttgtaccctcacggtctgcttgctggcttggctcgccttccccgatgtgcagggacagtgcgccaac |
| Equine Herpesvirus 1 | Ctctggaggatgaagctcttctgagtgcgatcgtagcggcggctcatggccaccgaggcggccgcgtgtg |
|  | Tgcgatcgtagcggcggctcatggccaccgaggcggccgcgtgtggcagggcccagagcgcgttcccggc |
|  | Cgccccggcaccatccgcctggtcgtcctcgtccatcgaggacgaggacgaggacgacgatgagatggag |
| Equine Herpesvirus 2 | Cgggttctacaacaccccggtgctggtggtggactttgccagcctgtaccccaccatcatccaggcccac |
|  | Gtggtggactttgccagcctgtaccccaccatcatccaggcccacaacctctgctactccaccatgatcc |
| Bovine Herpesvirus 1 | Cctcctcgtcctcgtcctcgtccgcggcgtcgtcccccgcctcgtcggacgacgacgaggccggcgccgc |
|  | Tccgtccattggctcggtcacgaagagcgacattgactggctctggtccacgaacggggcgcggtcggcg |
|  | Agcgacattgactggctctggtccacgaacggggcgcggtcggcgcagaggtcgatcaggcgctcttggt |
| Alcelaphine Herpesvirus 1 | Gcactcgtcaataaagttggcaattgccactgcggagtcttggctgctgttcccctctatagttatttga |
|  | Gccactgcggagtcttggctgctgttcccctctatagttatttgagcgcattttatcatgggatggatta |
| Saimiriine Herpesvirus 2 | Cagttgtgtttaacttacctccaagattaatagcagaatacgaagaatggcacaagtctcccatgtcatc |
|  | Attaatagcagaatacgaagaatggcacaagtctcccatgtcatcttatgtgtcaacgtgctcccagact |
| Ateline Herpesvirus 3 | Tgttaggtttggtctcattatttcttctttattagagacctttgatgacatgttagcaaaatacgggtaa |
|  | Gttatttttaaacagcatatttaaaatgttagcatacacctttccatgaatgttttccatggccatttgt |
| Gallid Herpesvirus 2 | Tcaaagggctactgtttttttggttccgaggaggcatggaaaaacttggtttttggttccattgattgct |
|  | Ccatcctcctccattggcgattacgtgatttagtagcatgggcgtaaaaaccggctccgacccagctccc |
| Transmissible Gastroenteritis Virus | Ctcttatggtggtgcttcagtttgtatttattgcagatgccatgttgaacatcctgctattgatggatta |
| Nipah Virus | Gacagtgatctgagtatgtatatgaaagataaagctttatctccaatcaaagacgaatgggacagtgtat |
|  | Tacacccgaaatggttctaatgtattgtgatgtcctagagggaaggatgatgatggagacaacagtcaaa |
| Equine Arthritis Virus | Cctatccacgcagggttttgtgttgcctagtgttttctccatggtgcgggcgtacttaaaagaggagatt |
|  | Tactgttccaaatacaagattaggagcattctgggcaccaacaattacattggcctaggtttgcgtgcct |
| Influenza C Virus | Gagcccagctcgatccctattgggaaatgaaacaccagatattgacaagactgaagcttatatgctctcg |
| Human Respiratory Syncytial Virus | Tagatagaattgatgaaaaattaagtgaaatattaggaatgctccatacattagtagttgcaagtgcagg |
|  | Tacaaagaaacaatagaaacatttgataacaatgaagaagaatctagctactcatatgaagagataaatg |
| Murine Hepatitis Virus | Ataattatgataagagtgctggctatccatttaataaatttggaaaggccaggctctattatgaggcatt |
|  | Gtactcatgggttgggactatcctaaatgtgatcgtgctatgccaaacatactgcgtattgttagtagtt |
| Hendra Virus | Ggagttaaaattagacagtgatttgagcatgtatatgaaagataaagcactatccccaataaaggaggaa |
| Poliovirus | Aagggagagttcactatgttaggagtccacgacaacgtggctattttaccaacccacgcttcacctggtg |
| Human Adenovirus C | Ccgctttccaagatggctaccccttcgatgatgccgcagtggtcttacatgcacatctcgggccaggacg |
|  | Gggggaccccatggcatggggtgggtgagcgcggaggcgtacatgccgcaaatgtcgtaaacgtagaggg |
| Human Rhinovirus 1b | Actctgttattccggtaactttgtacgccattttccctccctccccatccttttacgtaacttacaactt |
|  | Ttaaaactgggtgtgggttgttcccactcacaccacccaatgggtgttgtactctgttattccggtaact |
| Respiratory Syncytial Virus | Tagataggattgatgaaaaattaagtgaaatactaggaatgcttcacacattagtagtagcgagtgcagg |
|  | Ggcaaatatggaaacatacgtgaacaaacttcacgaaggctccacatacacagctgctgttcaatacaat |
| Bovine Respiratory Syncytial Virus | Ctactcatatgatgagataaatgatcaaacaaatgataatataacagcaagactagataggatagatgaa |
|  | Gtacaatgtcatagaaaaagatgatgatcctgcatctctcacaatatgggttcctatgttccaatcatcc |
| Human Adenovirus D | Tgcgtgagggcggaggcgtacatgccgcaaatgtcgtaaacatagatgggctccgagaagatgccgatgt |
|  | Gcgaagatggccaccccctcgatgatgccgcagtgggcgtacatgcacatcgccgggcaggacgcctcgg |

**CHAPTER 7**

**SYSTEM TESTING**

**7.1 Unit Testing**

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE NAME** | **TEST PROCEDURE** | **PRE-CONDITION** | **EXPECTED RESULT** |
| Valid\_login\_admin | Enter the valid username and password | None | The control should be transferred to the corresponding screen i.e., Menu screen. |
| Valid\_login\_user | Enter the valid username and password | None | The control should be transferred to the corresponding screen i.e., Menu screen which contains only the disease identification module. |
| Invalild\_login | Enter an invalid username or password | None | Error message is displayed. |

**Login Module**

Table 7.1 Test case for login module

**Admin Login**

**Management Module**

Table 7.2 Test case for management module

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE NAME** | **TEST PROCEDURE** | **PRE-CONDITION** | **EXPECTED RESULT** |
| Master\_load | Load the appropriate content for the given patient id. | Select the load content button from the content\_master screen. | The content is loaded. |
| Master\_new | Create a new patient id. | Select the new button from the content\_master screen | New patient id is generated. |
| Master\_delete | Delete an existing patient record. | Select the delete button from the content\_master screen. | Existing patient is deleted. |
| Master\_delete\_Invalid | Enter the Invalid Patient id to delete. | Select the delete button from the content\_master screen. | Error message is displayed. i.e., No record is found to delete. |
| Master\_save | Save the patient record | Select the save button from the content\_master screen. | Created patient record is saved. |
| Master\_save\_Invalid | Save the patient record without any content | Select the save button from the content\_master screen. | Error message is displayed. i.e., No content is loaded. |
| Master\_search | Search the required patient record. | Select the search button from the content\_master screen. | Required patient record is searched. |
| Master\_Search\_Invalid | Enter the Invalid Patient Id | Select the search button from the content\_master screen. | Error message is displayed. i.e., No record is found. |
| Master\_close | Close the content\_master | Select the close button from the content\_master screen. | Content\_master is closed. |

**Compression Module**

Table 7.3 Test case for compression

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE NAME** | **TEST PROCEDURE** | **PRE-CONDITION** | **EXPECTED RESULT** |
| Comp\_Exist\_load | Enter the valid patient id. | Select the patient id from the combo box | Content is loaded. |
| Comp\_Exist\_gencompress | Load the DNA sequence of a particular patient. | Select the apply gencompress button. | Compression is performed. |
| Comp\_sca\_load | Enter the valid patient id. | Select the patient id from the combo box | Content is loaded. |
| Comp\_sca\_applysca | Load the DNA sequence of a particular patient. | Select the apply compress button | Compression is performed. |

**Disease Identification Module**

Table 7.4 Test case for disease identification

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE NAME** | **TEST PROCEDURE** | **PRE-CONDITION** | **EXPECTED RESULT** |
| Disease\_load | Enter the valid patient id. | Select the patient id from the combo box. | DNA sequence corresponding to the id is loaded. |
| Disease\_identify | The record containing particular disease is loaded. | Click the identify disease button. | Name of the virus causing the disease is displayed if found |
| Disease\_identify\_notfound | The record not containing any disease is loaded. | Click the identify disease button. | Disease not found message is displayed. |

**7.2 Integration Testing**

Table 7.5 Integration testing

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE NAME** | **TEST PROCEDURE** | **PRE-CONDITION** | **EXPECTED RESULT** |
| Valid\_login\_admin | Enter the valid username and password | None | The control should be transferred to the corresponding screen i.e., Menu screen. |
| Valid\_login\_user | Enter the valid username and password | None | The control should be transferred to the corresponding screen i.e., Menu screen which contains only the disease identification module. |
| Master\_load | Load the appropriate content for the given patient id. | Select the load content button from the content\_master screen. | The content is loaded. |
| Comp\_Exist\_gencompress | Load the DNA sequence of a particular patient. | Select the apply gencompress button. | Compression is performed. |
| Comp\_sca\_load | Enter the valid patient id. | Select the patient id from the combo box | Content is loaded. |
| Comp\_sca\_applysca | Load the DNA sequence of a particular patient. | Select the apply compress button | Compression is performed. |
| Disease\_load | Enter the valid patient id. | Select the patient id from the combo box. | DNA sequence corresponding to the id is loaded. |
| Disease\_identify | The record containing particular disease is loaded. | Click the identify disease button. | Name of the virus causing the disease is displayed if found |

**CHAPTER 8**

**SYSTEM IMPLEMENTATION**

**8.1 GUI Design**

**Login**

Accepts the user id and password. Validates the user id and password.

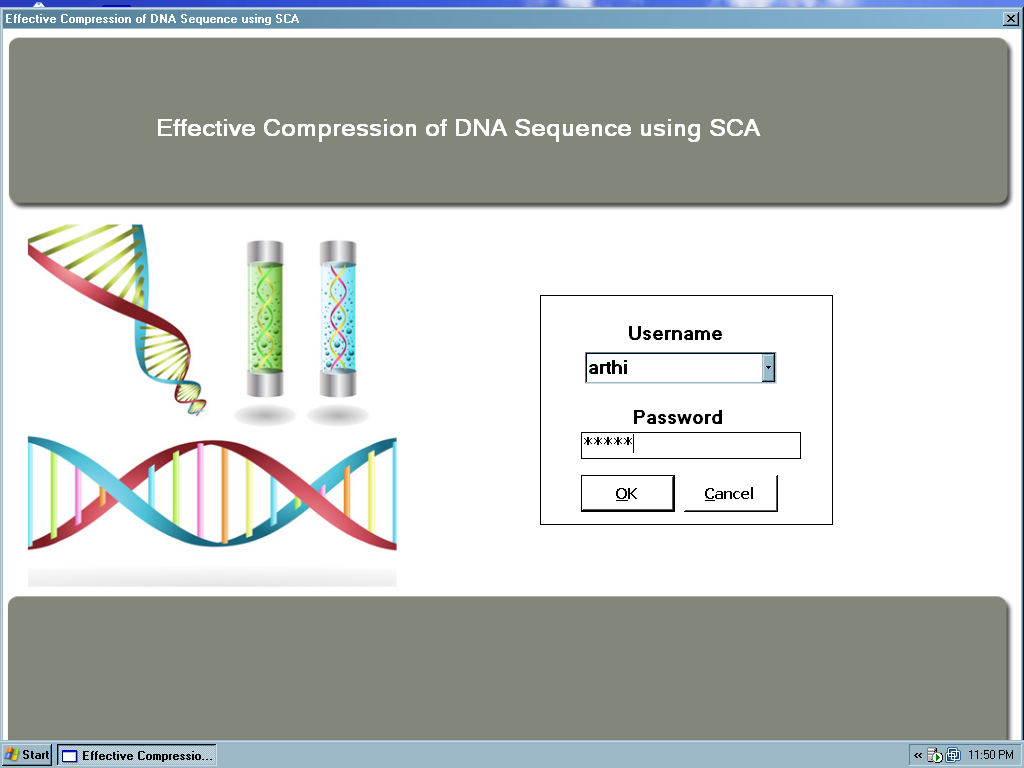


Figure 8.1 Login page for user and admin

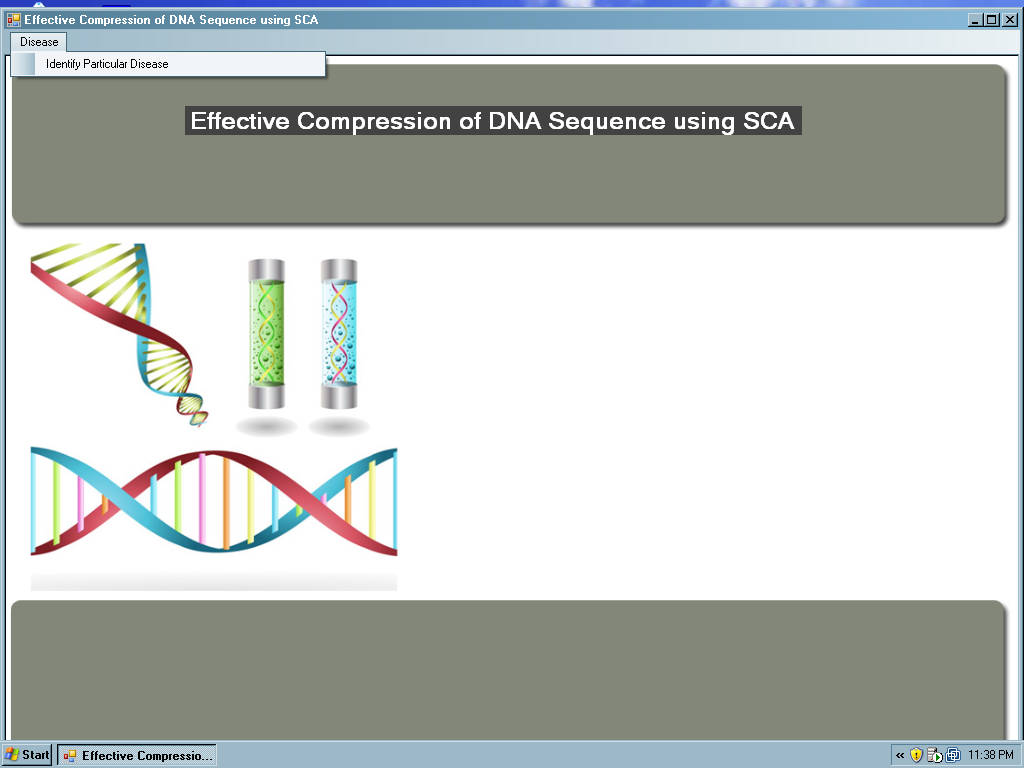
****

Figure 8.2 Module for user login

**Admin**

Chooses the desired option out of the three options like Genpack content module, DNA compression module, Disease identification module.

* If the option is Genpack content module then admin can load the patient’s DNA content.
* If the option is DNA compression module then the admin can compress the DNA sequence loaded from the database where it was stored primarily.
* If the option is disease identification module then the admin can identify the disease available in the DNA sequence loaded for the given id.

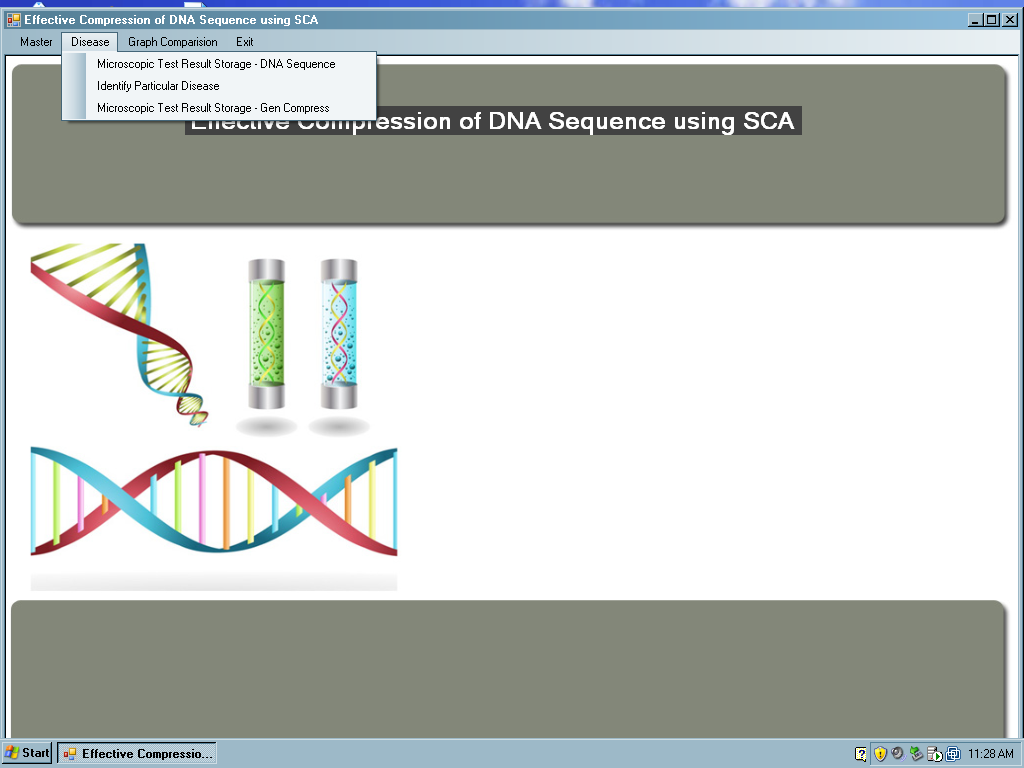
****

Figure 8.3 Modules for admin login

**Genpack Content**

This module is for the admin to load the desired contact. The admin is given several options like

* If the option is add then admin can add a new patient.
* If the option is delete then the admin can delete the patient from the database where it was stored primarily.
* If the option is search then the admin can search the contact by accessing the tables which has to be searched.
* If the option is save then the admin can save the loaded file of the given patient id.
* If the option is clear then the admin can clear contents that are not required.

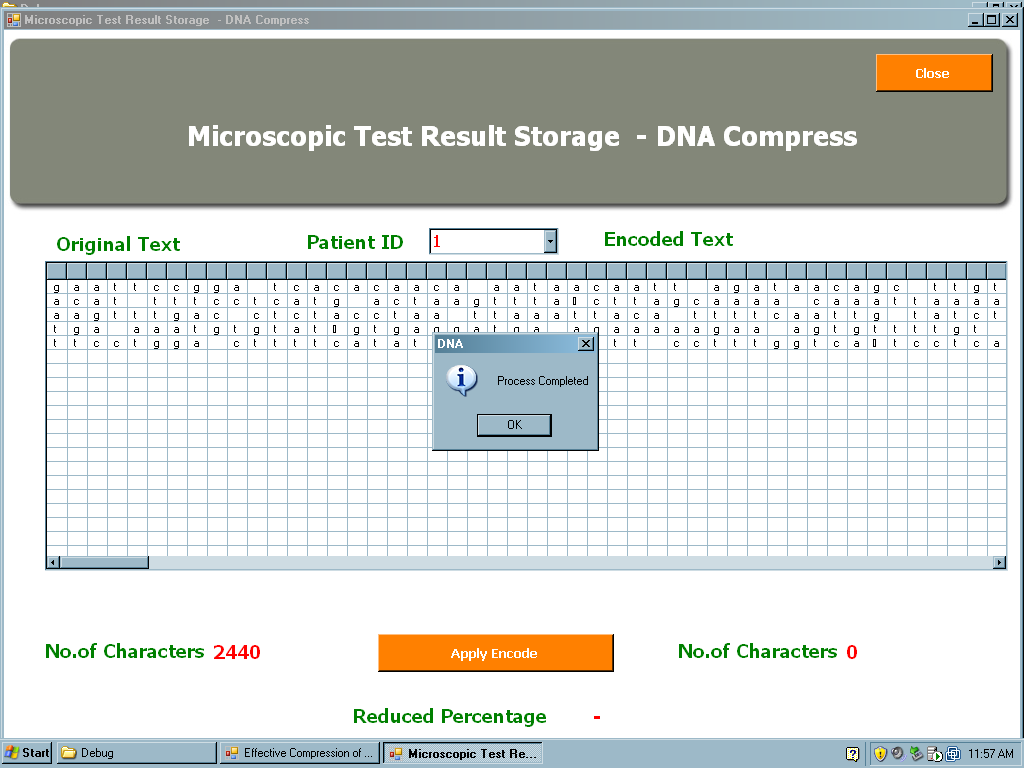
****

Figure 8.4 Contents loaded for compression

**DNA Compression**

This module is for the admin to compress the DNA content loaded for a particular patient. The file size and the reduced percentage are also displayed.

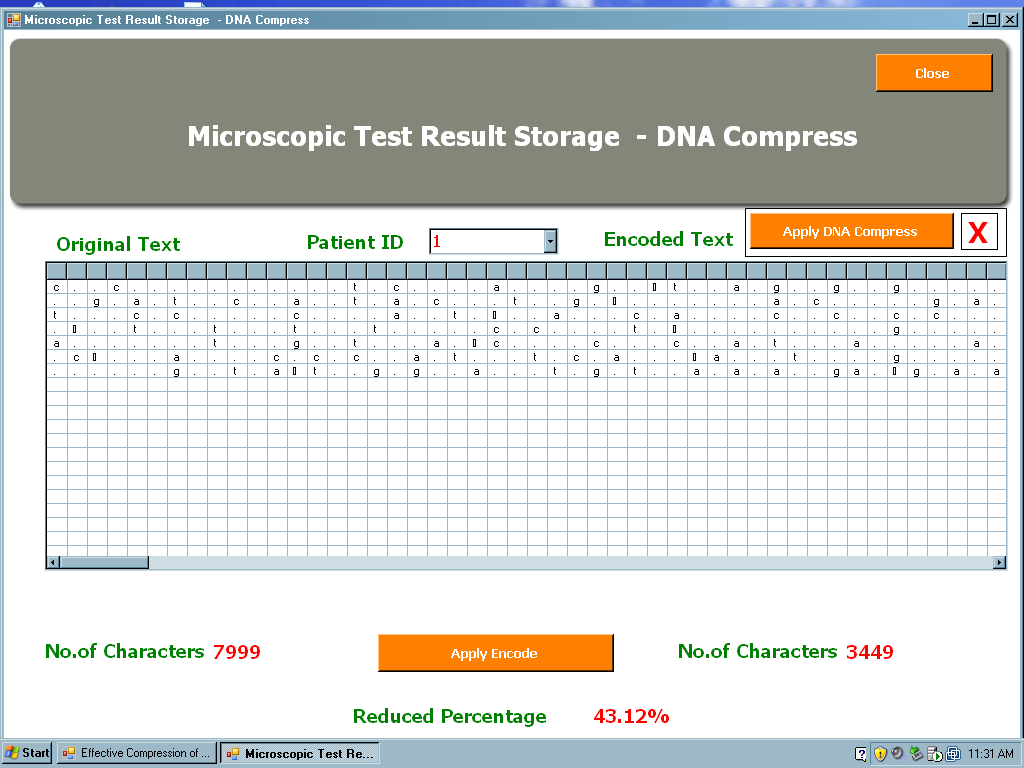


Figure 8.5 Result of compression

**Disease Identification**

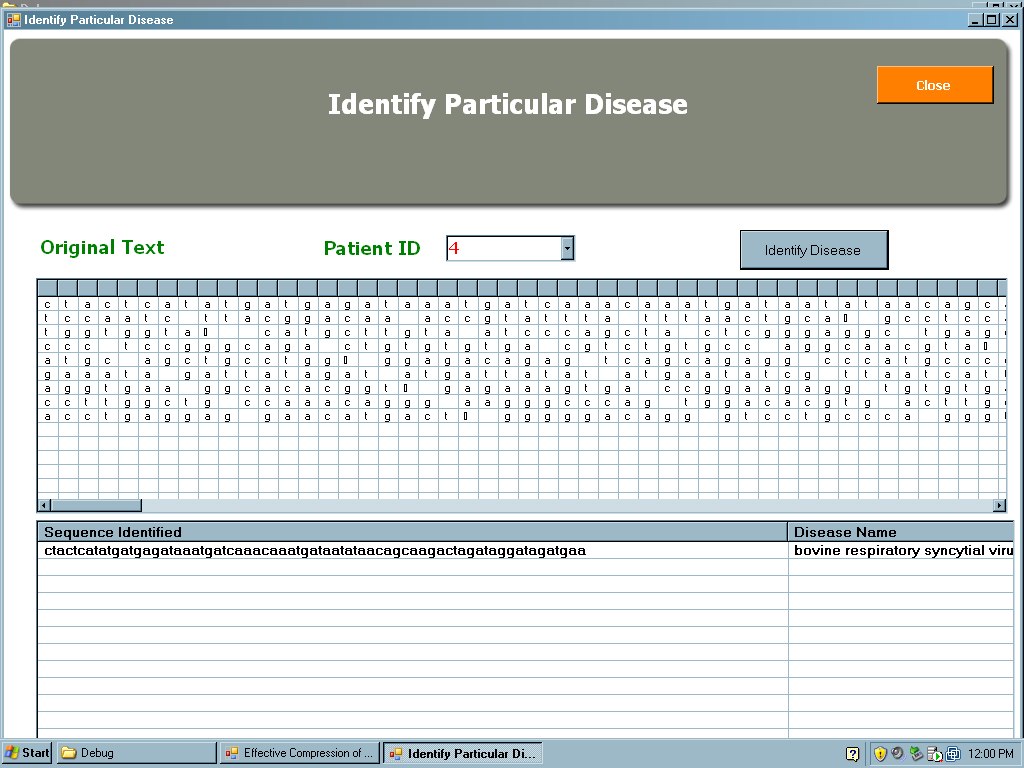


Figure 8.6 Output displaying the disease

**8.2 Graphical Comparison of Gencompress and SCA**

This graph shows a comparison of SCA and Gencompress algorithms. The DNA samples of seven patients were taken. These samples were compressed by using both the Gencompress and the SCA algorithms. The graph at the uppermost right side displays the Gencompress result and the graph at the bottom shows the compression result of Sequence Compression Algorithm. The graph at the centre shows the comparison of both the algorithms keeping patient id in the x-axis and the compression percentage in the y-axis.

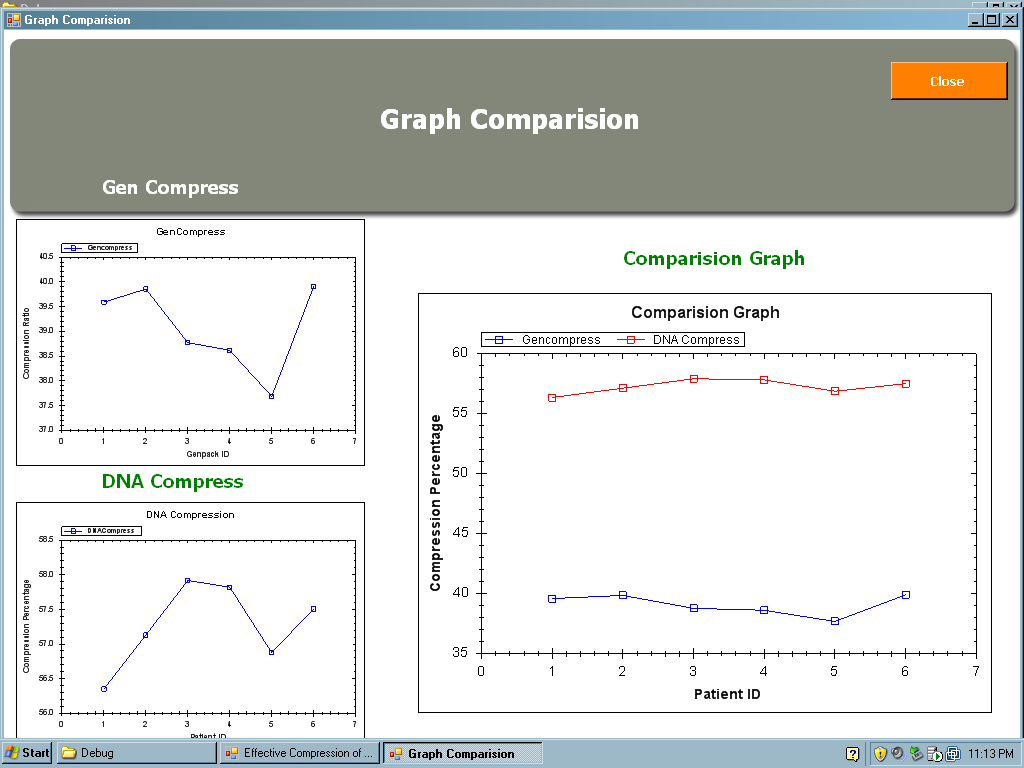
****

Figure 8.7 Comparison of Gencompress and SCA

**8.3 Percentage Comparison Of Gencompress And SCA**

This table shows the compression rate of the seven different samples, compressed using the Sequence Compression Algorithm and the Gencompress algorithm.

Table 8.1 Compression rate of SCA and GENCOMPRESS

|  |  |  |
| --- | --- | --- |
| **SAMPLE** | **GENCOMPRESS** | **SEQUENCE COMPRESSION** |
| 1 | 39.59% | 56.35% |
| 2 | 39.85% | 57.13% |
| 3 | 38.77% | 57.92% |
| 4 | 38.62% | 57.82% |
| 5 | 37.69% | 56.88% |
| 6 | 39.90% | 57.51% |

**CHAPTER 9**

**CONCLUSION AND FUTURE ENHANCEMENT**

**Conclusion**

Thus by using the “DNA Sequence Compression Algorithm” a better compression ratio was achieved when compared to other compression algorithms that are in use for genomic data. The file size has been compressed upto 60% and the algorithm was found to be very effective in comparison with the Gencompress, the best algorithm identified so far [12]. It has reduced the file size to an average of 58.98%. The system has greatly reduced the clerical overhead and drastically reduced the file size. The software used to develop the system makes it more flexible, portable and more secure. The process of disease identification was found to be very effective. The lookup table designed for disease identification contains the information about DNA sequences for various viruses collected from several sources.

**Future Enhancement**

The “Design of a DNA Sequence Compression Tool” has been developed according to the specifications mentioned. In future this system can be modified to suit the new requirements. The search option to load the data from popular genomic databases can be included and the compression of the same can be done using our project. Also the DNA sequence compression tool can be integrated with BLAST tool to make it more useful. DNA sequence compression tool can be made open source, making it available to everyone and to invite in new advancements.

**APPENDICES**

**APPENDIX 1 SOURCE CODE**

**SCA**

Imports System.IO

Public Class Frm\_testresult

Dim cn As New ADODB.Connection

Dim rs As New ADODB.Recordset

Dim dynaStrings() As String

Dim gen\_string() As String

Dim dystrings() As String

Dim s1, s2, s3, s4 As String

Dim space\_strings() As String

Dim i As Long

Dim j As Long

Dim temp As Integer

Dim z As Long

Dim perarr() As String

Dim ztring As String

Dim permutestring As String

Dim TmpStrArray() As String

Dim base As String

Private Sub Frm\_encode\_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

cn = New ADODB.Connection

cn.Open("Provider=SQLOLEDB;Data Source=.;Integrated Security=SSPI;Initial Catalog=Genpack")

'cn.Open("Provider=SQLOLEDB;Data Source=.;Integrated Security=SSPI;Initial Catalog=Genpack")

Call load\_id()

pb1.Value = 0

pb1.Minimum = 0

pb1.Maximum = 0

End Sub

Public Sub load\_id()

cbo\_id.Items.Clear()

rs = New ADODB.Recordset

rs.Open("Select \* from gen\_master", cn, ADODB.CursorTypeEnum.adOpenDynamic, ADODB.LockTypeEnum.adLockOptimistic)

If rs.EOF = True Then Exit Sub

While Not rs.EOF

cbo\_id.Items.Add(rs.Fields(0).Value)

rs.MoveNext()

End While

rs.Close()

End Sub

'Calculate First Encoding stage

'txt\_encoding.Text = ""

'Dim l As Integer

'Dim cc As Integer

'cc = 0

'For l = 0 To lv\_encode.Items.Count - 1

' txt\_encoding.Text = txt\_encoding.Text & lv\_encode.Items(l).Text

' For cc = 0 To 499

' txt\_encoding.Text = txt\_encoding.Text & lv\_encode.Items(l).SubItems(cc).Text

' Next cc

Dim temp As String

txt\_encoding.Text = ""

temp = ""

Dim l As Integer

Dim cc As Integer

cc = 0

For l = 0 To dynaStrings.Length - 2

If dynaStrings(l) = " " Then

Else

temp = temp & dynaStrings(l)

End If

Next l

txt\_encoding.Text = temp

com\_char.Text = txt\_encoding.Text.Length

End Sub

Public Sub tran\_copy()

For i = 0 To dynaStrings.Length - 2

dystrings(i) = dynaStrings(i)

Next i

End Sub

Public Sub tetra\_compress()

Dim t1, t2, t3 As Integer

Dim s1, s2, s3, s4 As String

Dim temp As String

s1 = ""

s2 = ""

s3 = ""

s4 = ""

txt\_encoding.Text = ""

temp = ""

Dim l As Integer

Dim temp\_l As Integer

Dim cc As Integer

cc = 0

For l = 0 To dystrings.Length - 2

temp\_l = l

If dystrings(l) = " " Then

Else

s1 = dystrings(l)

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

a: If dystrings(l) = " " Then

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

GoTo a

Else

s2 = dystrings(l)

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

End If

b: If dystrings(l) = " " Then

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

GoTo b

Else

s3 = dystrings(l)

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

End If

c: If dystrings(l) = " " Then

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

GoTo c

Else

s4 = dystrings(l)

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

End If

l = temp\_l

End If

If cmp\_str = s1 & s2 & s3 & s4 Then

s1 = ""

s2 = ""

s3 = ""

s4 = ""

If temp\_l >= dystrings.Length - 3 Then

GoTo d End If

dystrings(temp\_l) = "."

dystrings(temp\_l + 1) = " "

dystrings(temp\_l + 2) = " "

dystrings(temp\_l + 3) = " "

End If

Next l

lv\_encode.Items.Clear()

Dim flag As Boolean

flag = False

Dim arr As Integer

Dim col, row As Integer

col = 0 : row = 0

For arr = 0 To dystrings.Length - 2

If dystrings(arr) = " " Then

ElseIf flag = False And dystrings(arr) <> " " Then

lv\_encode.Items.Add(dystrings(arr))

flag = True

ElseIf flag = True And dystrings(arr) <> " " Then

If col = 499 Then

col = 0

flag = False

row = row + 1

Else

lv\_encode.Items(row).SubItems.Add(dystrings(arr))

col = col + 1

End If

End If

Next arr

d: Dim temp1 As String

txt\_encoding.Text = ""

temp = ""

Dim l\_1 As Integer

Dim cc1 As Integer

cc = 0

For l\_1 = 0 To dystrings.Length - 2

If dystrings(l) = " " Then

Else

temp1 = temp1 &dystrings(l\_1)

End If

Next l\_1

Call random\_clear()

End Sub

Public Sub random\_clear()

Dim temp As String

txt\_encoding.Text = ""

temp = ""

Dim l As Integer

Dim cc As Integer

cc = 0

For l = 0 To dystrings.Length - 2

If dystrings(l) = " " Then

Else

temp = temp & dystrings(l)

End If

Next l

txt\_encoding.Text = temp

com\_char.Text = txt\_encoding.Text.Length

End Sub

Public Sub tri\_compress()

Dim t1, t2, t3 As Integer

Dim s1, s2, s3 As String

Dim temp As String

s1 = ""

s2 = ""

s3 = ""

txt\_encoding.Text = ""

temp = ""

Dim l As Integer

Dim temp\_l As Integer

Dim cc As Integer

cc = 0

For l = 0 To dystrings.Length - 2

temp\_l = l

If dystrings(l) = " " Then

Else

s1 = dystrings(l)

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

a: If dystrings(l) = " " Then

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

GoTo a

Else

s2 = dystrings(l)

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

End If

b: If dystrings(l) = " " Then

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

GoTo b

Else

s3 = dystrings(l)

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

End If

l = temp\_l

End If

If cmp\_str = s1 & s2 & s3 Then

s1 = ""

s2 = ""

s3 = ""

If temp\_l >= dystrings.Length - 3 Then

GoTo d

End If

dystrings(temp\_l) = "."

dystrings(temp\_l + 1) = " "

dystrings(temp\_l + 2) = " "

End If

Next l

lv\_encode.Items.Clear()

Dim flag As Boolean

flag = False

Dim arr As Integer

Dim col, row As Integer

col = 0 : row = 0

For arr = 0 To dystrings.Length - 2

If dystrings(arr) = " " Then

ElseIf flag = False And dystrings(arr) <> " " Then

lv\_encode.Items.Add(dystrings(arr))

flag = True

ElseIf flag = True And dystrings(arr) <> " " Then

If col = 499 Then

col = 0

flag = False

row = row + 1

Else

lv\_encode.Items(row).SubItems.Add(dystrings(arr))

col = col + 1

End If

End If

Next arr

d: Dim temp1 As String

txt\_encoding.Text = ""

temp = ""

Dim l\_1 As Integer

Dim cc1 As Integer

cc = 0

For l\_1 = 0 To dystrings.Length - 2

If dystrings(l) = " " Then

Else

temp1 = temp1 &dystrings(l\_1)

End If

Next l\_1

Call random\_clear()

End Sub

Public Sub bi\_compress()

Dim t1, t2, t3 As Integer

Dim s1, s2 As String

Dim temp As String

s1 = ""

s2 = ""

txt\_encoding.Text = ""

temp = ""

Dim l As Integer

Dim temp\_l As Integer

Dim cc As Integer

cc = 0

For l = 0 To dystrings.Length - 2

temp\_l = l

If dystrings(l) = " " Then

Else

s1 = dystrings(l)

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

a: If dystrings(l) = " " Then

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

GoTo a

Else

s2 = dystrings(l)

l = l + 1

If l >= dystrings.Length - 2 Then

GoTo d

End If

End If

l = temp\_l

End If

If cmp\_str = s1 & s2 Then

s1 = ""

s2 = ""

If temp\_l >= dystrings.Length - 3 Then

GoTo d

End If

dystrings(temp\_l) = "."

dystrings(temp\_l + 1) = " "

End If

Next l

**Disease Identification**

Imports System.IO

Public Class Frm\_disease

Dim cn As New ADODB.Connection

Dim rs As New ADODB.Recordset

Dim dynaStrings() As String

Dim gen\_string() As String

Dim dystrings() As String

Dim s1, s2, s3, s4 As String

Dim space\_strings() As String

Dim i As Long

Dim j As Long

Dim temp As Integer

Dim z As Long

Dim perarr() As String

Dim ztring As String

Dim permutestring As String

Dim TmpStrArray() As String

Dim base As String

Public t1, t2, t3 As Integer

Private Sub Frm\_color\_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

cn = New ADODB.Connection

'cn.Open("Provider=SQLOLEDB;Data Source=.;Integrated Security=SSPI;Initial Catalog=Genpack")

cn.Open("Provider=SQLOLEDB;Data Source=.;Integrated Security=SSPI;Initial Catalog=Genpack")

Call load\_id()

Call load\_sym()

End Sub

Public Sub load\_sym()

lv2.Items.Clear()

rs = New ADODB.Recordset

rs.Open("Select \* from sym\_table", cn, 1, 2)

If rs.EOF = True Then

Else

Dim i As Integer

i = 0

While Not rs.EOF

lv2.Items.Add(rs.Fields(0).Value)

lv2.Items(i).SubItems.Add(rs.Fields(1).Value)

lv2.Items(i).SubItems.Add(rs.Fields(2).Value)

i = i + 1

rs.MoveNext()

End While

End If

rs.Close()

End Sub

Public Sub space\_in\_array()

On Error Resume Next

j = 0

For i = 0 To dynaStrings.Length - 2

If dynaStrings(i) = " " Then

space\_strings(j) = i

j = j + 1

End If

Next

calc\_space = j

Dim l As Long

l = dynaStrings.Length - 1

l = 0

Dim z As Integer

For z = 0 To lv\_encode.Items.Count - 1

l = l + 1

If lv\_encode.Items(z).Text = " " Then

lv\_encode.Items(z).UseItemStyleForSubItems = False

lv\_encode.Items(z).BackColor = Color.Yellow

Dim k As Integer

For k = 0 To 499

l = l + 1

If l >= dynaStrings.Length - 1 Then

Exit Sub

End If

If lv\_encode.Items(z).SubItems(k).Text = " " Then

lv\_encode.Items(z).UseItemStyleForSubItems = False

lv\_encode.Items(z).SubItems(k).BackColor = Color.Yellow

End If

Next k

Else

Dim k As Integer

For k = 0 To 499

l = l + 1

If l >= dynaStrings.Length - 1 Then

Exit Sub

End If

If lv\_encode.Items(z).SubItems(k).Text = " " Then

lv\_encode.Items(z).UseItemStyleForSubItems = False

lv\_encode.Items(z).SubItems(k).BackColor = Color.Yellow

End If

Next k

End If

Next z

End Sub

Public Sub split\_chars\_into\_array()

For i = 0 To dynaStrings.Length - 2

dynaStrings(i) = txt\_original.Text.Substring(i, 1)

Next (i)

lv\_encode.Clear()

Dim k As Integer

For k = 0 To 499

lv\_encode.Columns.Add("", 20)

Next

Dim r As Integer

r = 0

Dim z As Long

For z = 0 To dynaStrings.Length - 1

a: If z >= dynaStrings.Length - 1 Then

Exit Sub

End If

lv\_encode.Items.Add(dynaStrings(z))

For k = 0 To 499

If z >= dynaStrings.Length - 1 Then

Exit Sub

End If

z = z + 1

lv\_encode.Items(r).SubItems.Add(dynaStrings(z))

If k = 499 Then

If z >= dynaStrings.Length - 1 Then

Exit Sub

End If

k = 0

r = r + 1

z = z + 1

'MsgBox(z)

GoTo a

End If

Next k

Next z

End Sub

Public Sub create\_array()

ReDim dynaStrings(txt\_original.Text.Length)

ReDim dystrings(dynaStrings.Length)

ReDim space\_strings(txt\_original.Text.Length)

End Sub

Public Sub tran\_copy()

For i = 0 To dynaStrings.Length - 2

dystrings(i) = dynaStrings(i)

Next i

End Sub

Public Sub random\_clear()

Dim temp As String

txt\_encoding.Text = ""

temp = ""

Dim l As Integer

Dim cc As Integer

cc = 0

For l = 0 To dystrings.Length - 2

If dystrings(l) = " " Then

Else

temp = temp & dystrings(l)

End If

Next l

txt\_encoding.Text = temp

com\_char.Text = txt\_encoding.Text.Length

End Sub

Private Sub Button2\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click

End Sub

Public Sub call\_apply()

rs = New ADODB.Recordset

rs.Open("select \* from apply\_color", cn, 1, 2)

Dim t As String

t = ""

If chk\_a.Checked = True Then

t = "a"

ElseIf chk\_c.Checked = True Then

t = "c"

ElseIf chk\_t.Checked = True Then

t = "t"

ElseIf chk\_g.Checked = True Then

t = "g"

End If

If ch\_a.Checked = True Then

t = t + "a"

ElseIf ch\_c.Checked = True Then

t = t + "c"

ElseIf ch\_t.Checked = True Then

t = t + "t"

ElseIf ch\_g.Checked = True Then

t = t + "g"

End If

rs.Fields(0).Value = t

rs.Fields(1).Value = cbo\_color.Text

rs.Update()

'MsgBox(cbo\_color.Text & " : Color Set Successfully for - " & t)

rs.Close()

End Sub

Private Sub Button5\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button5.Click

Call load\_sym()

Call show\_unwanted()

Call remove\_unwanted()

Call call\_apply()

rs = New ADODB.Recordset

rs.Open("Select \* from apply\_color", cn, 1, 2)

If rs.EOF = True Then Exit Sub

Dim j, k As Integer

j = 0

k = 0

For j = 1 To lv\_encode.Items.Count - 1

For k = 2 To lv\_encode.Items(j).SubItems.Count - 1

lv\_encode.Items(j).UseItemStyleForSubItems = False

lv\_encode.Items(j).SubItems(k - 1).BackColor = Color.White

lv\_encode.Items(j).SubItems(k).BackColor = Color.White

Next k

Next j

For j = 1 To lv\_encode.Items.Count - 1

For k = 2 To lv\_encode.Items(j).SubItems.Count - 1

If lv\_encode.Items(j).SubItems(k - 1).Text & lv\_encode.Items(j).SubItems(k).Text = Trim(rs.Fields(0).Value) Then

lv\_encode.Items(j).UseItemStyleForSubItems = False

lv\_encode.Items(j).SubItems(k - 1).BackColor = Color.Red

lv\_encode.Items(j).SubItems(k).BackColor = Color.Red

End If

Next k

Next j

rs = New ADODB.Recordset

If chk\_a.Checked = True And ch\_a.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'A' and to\_col ='A'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "A" And lv2.Items(i).SubItems(1).Text = "A" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

End If

rs.Close()

ElseIf chk\_a.Checked = True And ch\_c.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'A' and to\_col ='C'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "A" And lv2.Items(i).SubItems(1).Text = "C" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

ElseIf chk\_a.Checked = True And ch\_t.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'A' and to\_col ='T'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "A" And lv2.Items(i).SubItems(1).Text = "T" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

ElseIf chk\_a.Checked = True And ch\_g.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'A' and to\_col ='G'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "A" And lv2.Items(i).SubItems(1).Text = "G" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

ElseIf chk\_c.Checked = True And ch\_a.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'C' and to\_col ='A'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "C" And lv2.Items(i).SubItems(1).Text = "A" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

ElseIf chk\_c.Checked = True And ch\_c.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'C' and to\_col ='C'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "C" And lv2.Items(i).SubItems(1).Text = "C" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

ElseIf chk\_c.Checked = True And ch\_t.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'C' and to\_col ='T'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "C" And lv2.Items(i).SubItems(1).Text = "T" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

ElseIf chk\_c.Checked = True And ch\_g.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'C' and to\_col ='G'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "C" And lv2.Items(i).SubItems(1).Text = "G" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

ElseIf chk\_t.Checked = True And ch\_a.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'T' and to\_col ='A'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "T" And lv2.Items(i).SubItems(1).Text = "A" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

ElseIf chk\_t.Checked = True And ch\_c.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'T' and to\_col ='C'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "T" And lv2.Items(i).SubItems(1).Text = "C" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

ElseIf chk\_t.Checked = True And ch\_t.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'T' and to\_col ='T'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "T" And lv2.Items(i).SubItems(1).Text = "T" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

ElseIf chk\_t.Checked = True And ch\_g.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'T' and to\_col ='G'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "T" And lv2.Items(i).SubItems(1).Text = "G" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

ElseIf chk\_g.Checked = True And ch\_a.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'G' and to\_col ='A'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

ElseIf chk\_g.Checked = True And ch\_c.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'G' and to\_col ='C'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "G" And lv2.Items(i).SubItems(1).Text = "C" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

ElseIf chk\_g.Checked = True And ch\_t.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'G' and to\_col ='T'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "G" And lv2.Items(i).SubItems(1).Text = "T" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

ElseIf chk\_g.Checked = True And ch\_g.Checked = True Then

rs.Open("Select \* from sym\_table where from\_col = 'G' and to\_col ='G'", cn, 1, 2)

If rs.EOF = True Then

Else

txt\_symptom.Text = rs.Fields(2).Value

End If

rs.Close()

Dim i As Integer

For i = 0 To lv2.Items.Count - 1

If lv2.Items(i).Text = "G" And lv2.Items(i).SubItems(1).Text = "G" Then

lv2.Items(i).BackColor = Color.GreenYellow

End If

Next i

Exit Sub

End If

End Sub

Private Sub cbo\_color\_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cbo\_color.SelectedIndexChanged

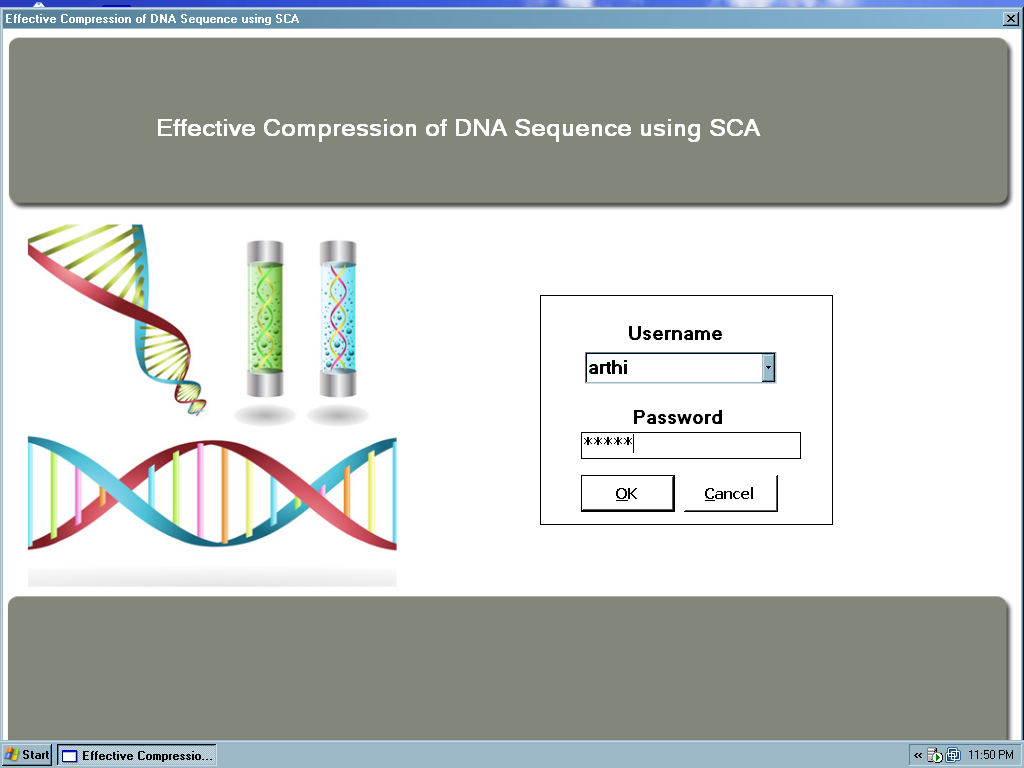
End Sub

End Class

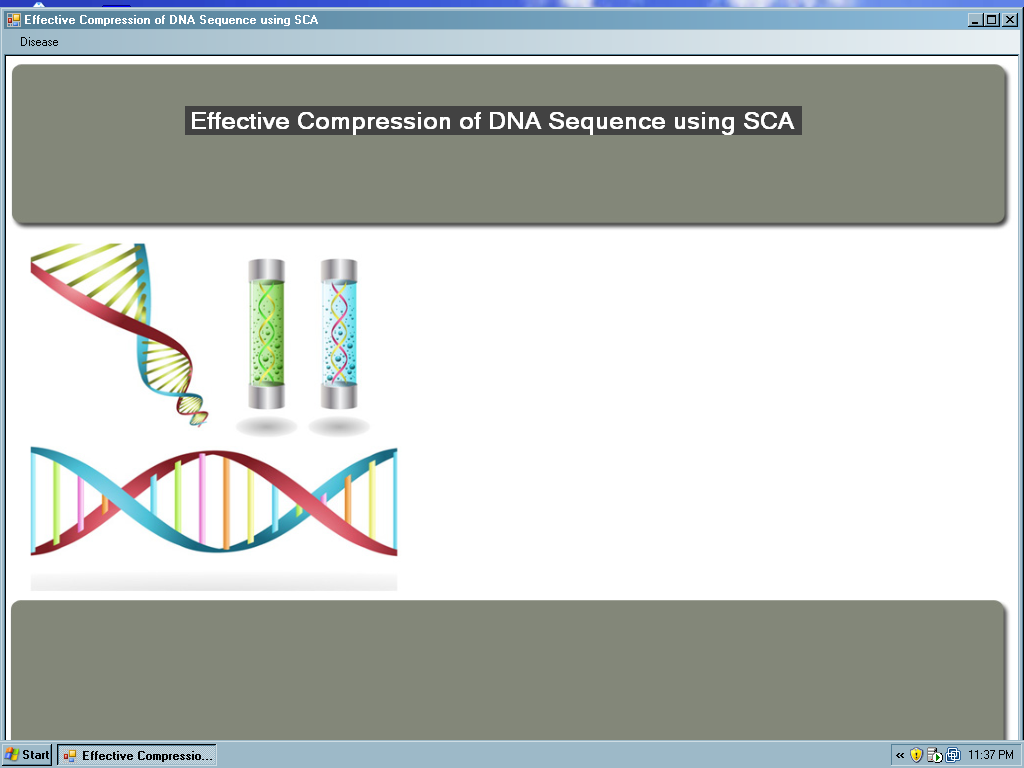
**APPENDIX 2**

**SCREEN SHOTS**

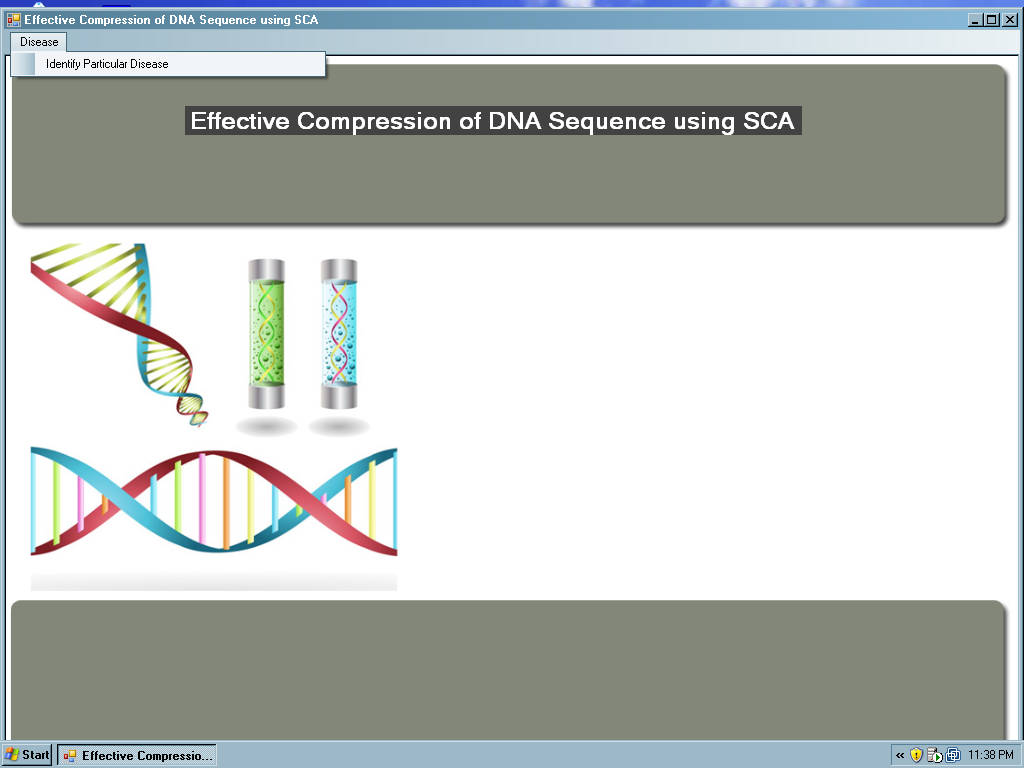
This shows the login screen for the user and the admin.

****

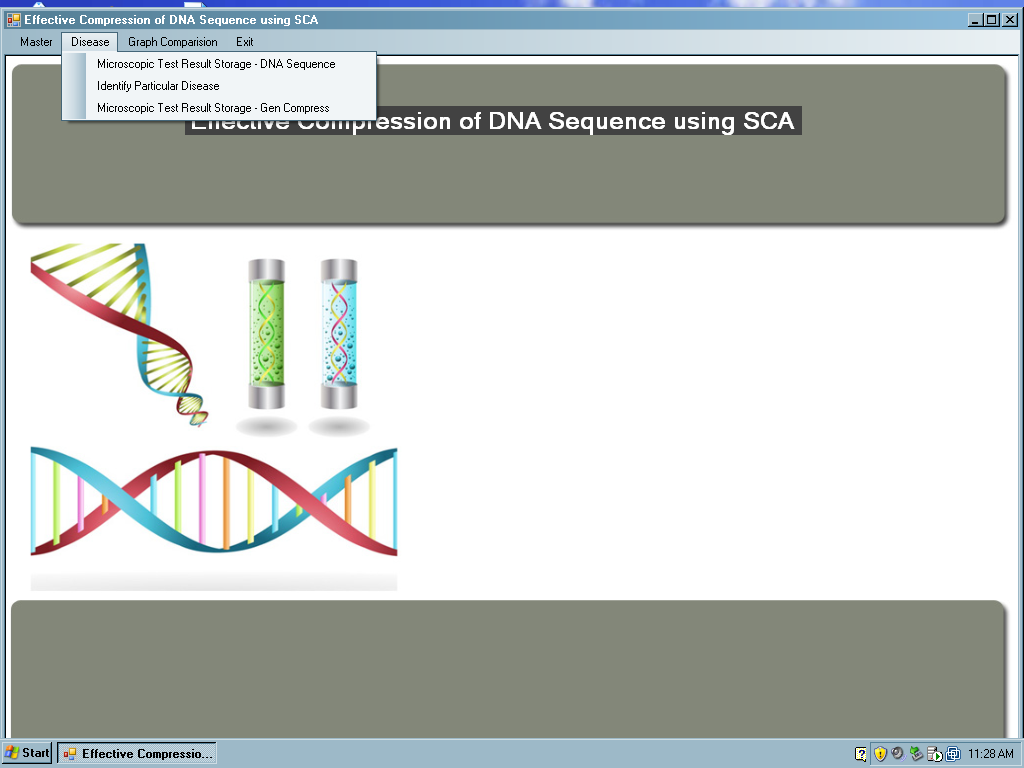
When the user login is given, the menu screen with disease identification module is displayed.

****

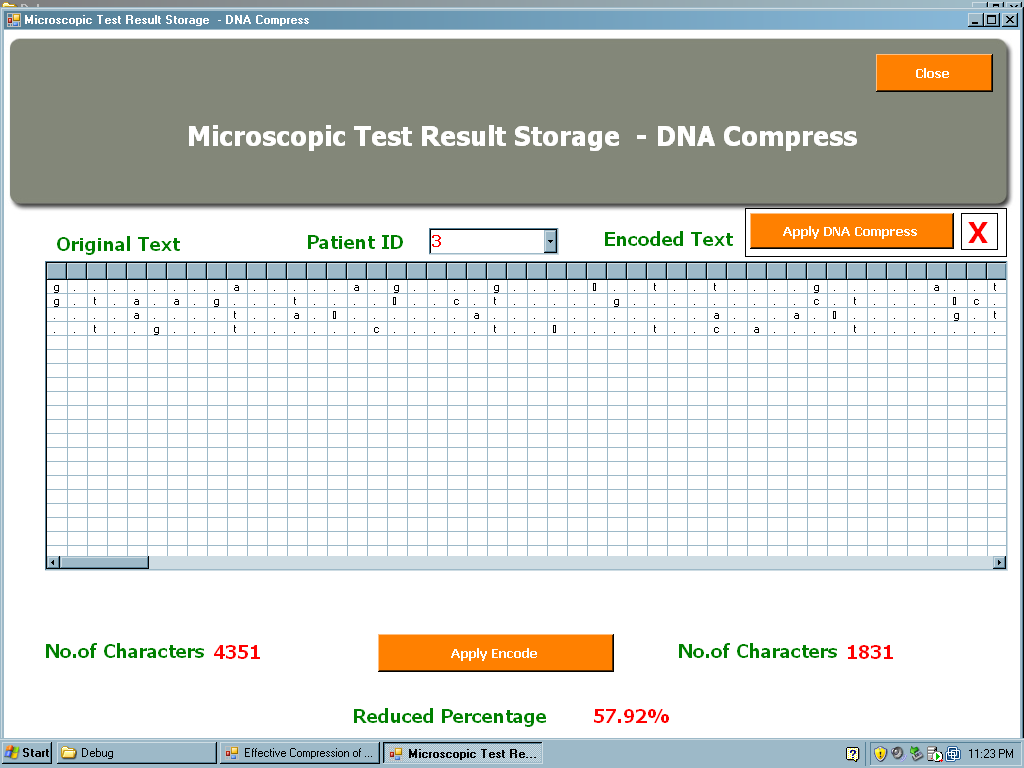
On selecting the disease option the user can identify the disease of a particular patient.

****

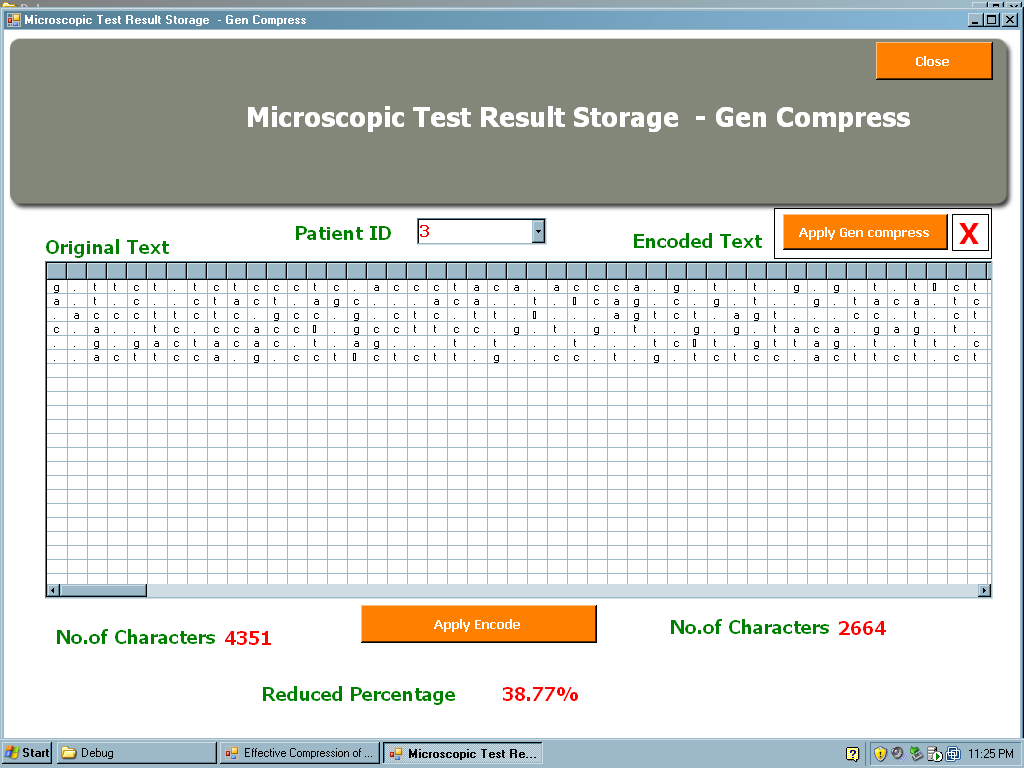
When the admin login is given, menu screen with all the modules is displayed.

****

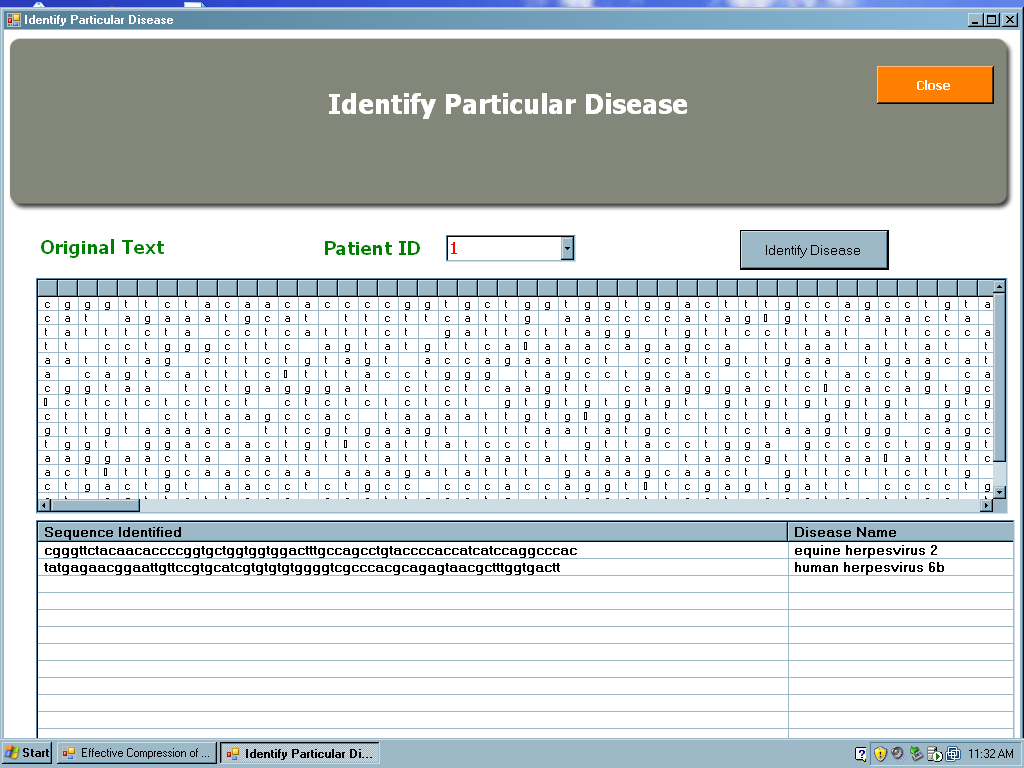
After entering the patient id and on selecting DNA compress, the compression starts using the SCA.

****

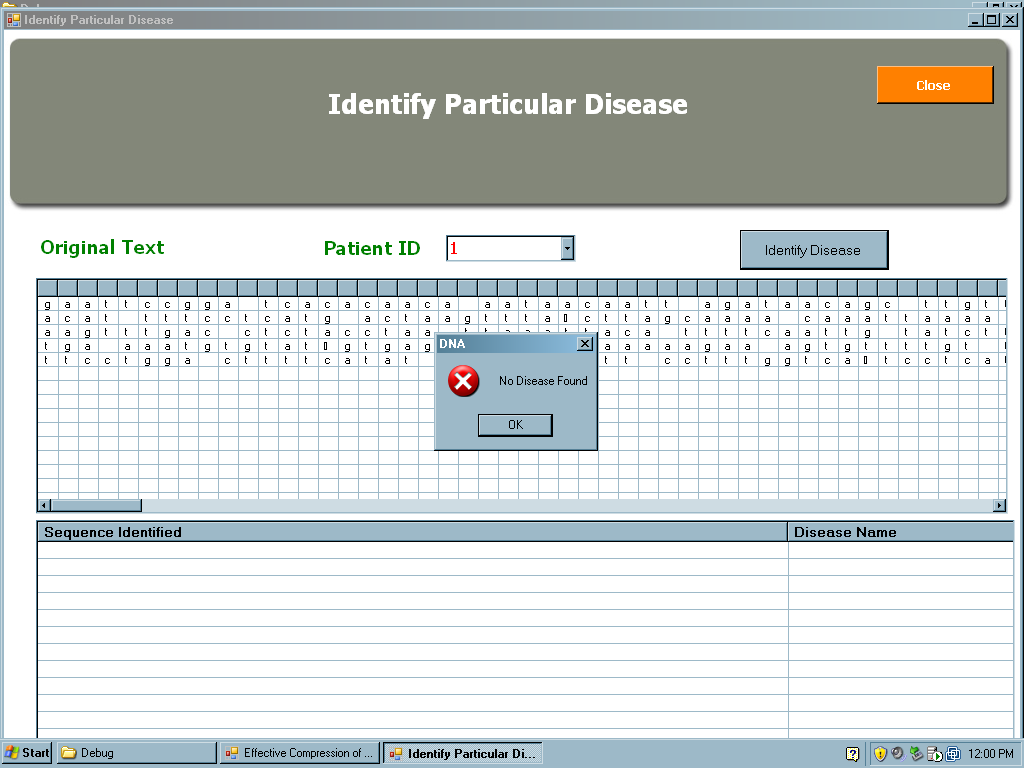
After entering the patient id and on selecting DNA compress, the compression starts using the SCA.

****

After entering the patient id and on selecting identify disease, the DNA sequence along with the virus name is displayed.

****

When the disease is not found, the error message is displayed.

****

**REFERENCES**

1. Chen. X, Li. M, Ma. B, and Tromp. J, “DNACompress: Fast and effective DNA sequence compression,” Bioinformatics, volume 18, number 12, page preferred 1696–1698, 2005.
2. Chen. X, Kwong. S, and Li. M, A compression algorithm for DNA sequences and its applications in genome comparison. recombination, page 107, 2000.
3. Korodi. G and Tabus. I, “An efficient normalized maximum likelihood algorithm for DNA sequence compression,” ACM Transcription Information System, volume 23, number 1, page preferred 3–34, 2005.
4. Rissanen. J, “Strong optimality of the normalized ML models as universal

codes and information in data,” IEEE Transcription Information Theory, volume 47, number 5, page preferred 1712–1717, 2001.

1. Rivals. E, Delahaye. J. P, Dauchet. M, and Delgrange. O, “A guaranteed compression Scheme for repetitive DNA sequences,” LIFL Lille I University, Lille, France, Technical IT–285, 2009.
2. Scott. D. Kahn, On the future of Genomic Data, Science Magazine, 11 February 2011
3. Stern. L, Allison. T, Edgoose, and T.I. Dix, “Sequence complexity for biological sequence analysis,” Computers & Chemistry, volume 24, page preffered 43-55, 2000.
4. Tabus, G. Korodi and J. Rissanen. “DNA sequence compression using the Normalized maximum likelihood model for discrete regression,” in Proceedings Data Compression Conference, page preferred 253–262, 2006.
5. Tabus. T and Korodi. G, “An efficient normalized maximum likelihood algorithm for DNA sequence compression,” ACM Transcriptio. Information System, volume 23, number 1, page preferred 3–34, 2008.
6. Toshiko Matsumoto, Kunihiko Sadakane, Hiroshi Imai, “Biological Sequence Compression Algorithm”, Genome Informatics volume 11, page preferred 43–52, 2000.
7. Tromp. J, Ma. B, and Li. M, “PatternHunter: Faster and more sensitive homology Search,” Bioinformatics, volume 18, number 3, page preferred 440–445, 2002.
8. Xin Chen, Sam Kwong, Ming Li, “A Compression Algorithm for DNA Sequences and Its Applications in Genome Comparison”, volume 11, page preferred 23-56, 2009.
9. <http://www.koshland-science-museum.org/exhibitdna/inf03activity.jsp>
10. <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1762815/>
11. <http://www.kaloramainformation.com/Generation-Sequencing-Survey-2657156/>