

**Name: Shawn Ronaldo Grant**

**Cand. #: 1000160537**

**Center #: 100016**

**Center: Campion College**

**ZENCYPHER**

**Text Encryption**

Computer Science Internal Assessment

**CAPE Computer Science Unit 1 Internal Assessment**

**TABLE of CONTENTS**

* Problem Definition
  + Background 2
  + Context Description 3
  + Problem Description 4
  + Objective Of Solution 5
* System Models
  + Structured Charts (HIPO) 6
  + Record Definitions 10
  + Data Structures 10
  + Dataflow Diagram 11
  + Entity Relation Diagram 12
* Design Specification
  + System Requirements
  + Interface Specifications
* Pseudocode
* C code

**PROBLEM DEFINITION**

**BACKGROUND**

Privacy of information has been an important part of history, from messages been sent secretly by birds to now, where we use the internet to send most of our information. Information sent over the internet can include: usernames, passwords, addresses and telephone numbers**.** With the rise of the internet, new threats to data security have arrived. All computer systems are susceptible to intrusion by hackers. New technology has been developed to combat the threat of hackers, but no system is hacker-proof yet.

Everyday people send sensitive data over the internet via social media or email. Although these networks may be strong against computer based attacks, no thought is put into the fact that once your phone or laptop is left open anyone can simply open your messages and view the data you’ve sent to others in plain text.

This is where local encryption is useful. Text and files can be encoded, so that even with access to the device it is impossible for anyone to make sense of the information. There are many different encryption algorithms, most popular being: MD5, RSA, SHA-1 and 3DES.

**CONTEXT DESCRIPTION**

Encryption is the process of encoding a message or information in such a way that only authorized parties can access it and those who are not authorized cannot.

When people send information over the internet, messages are encrypted and sent over a network to the recipient’s router, where it is decrypted and returned to the recipient’s device and represented as a plain text or file.

This works well for ensuring that external hackers cannot get intelligible data from information sent over the network.

This encryption, however, does not protect against anyone with access to the device from viewing messages as the data is only encoded while it is in transit.

If, say, you text a friend or relative your email password (for whatever reason), and either of you gets robbed and your phone stolen. There’s no way of protecting that message that was sent. The thief now has your email. Even if there is a lock on the device, hackers have always figured out ways of cracking those, since they are normally not very complex.

If the message sent was in an encoded form it would add an extra layer of protection on the information. There would be no way of determining what the original message was.

**PROBLEM DESCRIPTION**

* We share sensitive information regularly over plain text. This could be anything from a password to credit card information.
* Our information is only secure while it is being sent over a trusted network, by email or social media application.
* Many persons don’t have a simple screen lock on their phone, removing the first layer of security on a device
* The security of that information is comprised if someone simply takes your unlocked phone or computer (or you give it to them) and opens the email or messaging application you’ve already signed in to.
* A stolen device is the easiest way to get access to someone’s information.

**OBJECTIVE OF SOLUTION**

* Allow users to encode any text message to an unreadable form
* Allow users to create files with encoded messages inside
* Ensure that sensitive information stored on a device locally is protected against prying eyes
* Allow users to send an encrypted message and files to anyone without ever disclosing information.

**STRUCTURED CHART**

**DRIVER**

**SplashScreen**

**Decode**

**SaveToFile**

**Encode**

**MainMenu**

**LoginSignUp**

**strAppend**

**strEndsWith**

**strAppend**

**UpdateLogin**

**GoodBye**

**ShowHistorye**

**Clear**

**RECORD DEFINITION**

1. **TextCypher** Record 🡪 Contains information about the encoding

**Record TextCypher**

**original: String**

**encoded: String**

**dateTime: String**

**EndTextCypher**

1. **User** Record 🡪 Stores the current logged in user

**Record User**

**username: String**

**password: String**

**EndUser**

**DATA STRUCTURE**

A Stack was chosen as an ADT to store a collection of cypher text in the program.

A stack stores data in such a way that the last item inserted is the first to come out.

**Record Stack**

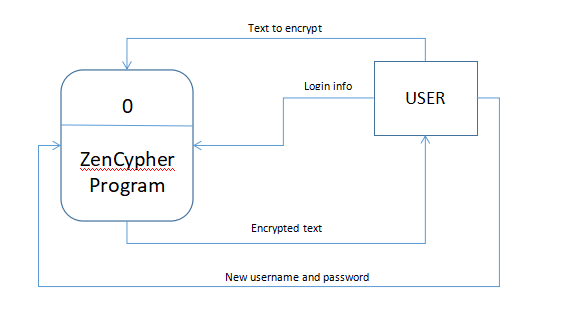
**Array storage: TextCypher [20]**

**top: Integer**

**EndStack**

**DATA FLOW DIAGRAM**

Context Diagram



Level 0 Diagram

D1 User Data File

**2**

**Accept Menu input**

Successful login

**3**

**Encoding Process**

English text

Encrypted text

**4**

**Decoding Process**

D2 History File

Decrypted text

Encrypted text

**5**

**Display history**

Decrypted text

Previous encryptions

**6**

**Update user login**

New credentials

D2 History File

User

**1**

**Login**

User credentials

Saved user credentials

Previous encryptions

Functional Requirements:

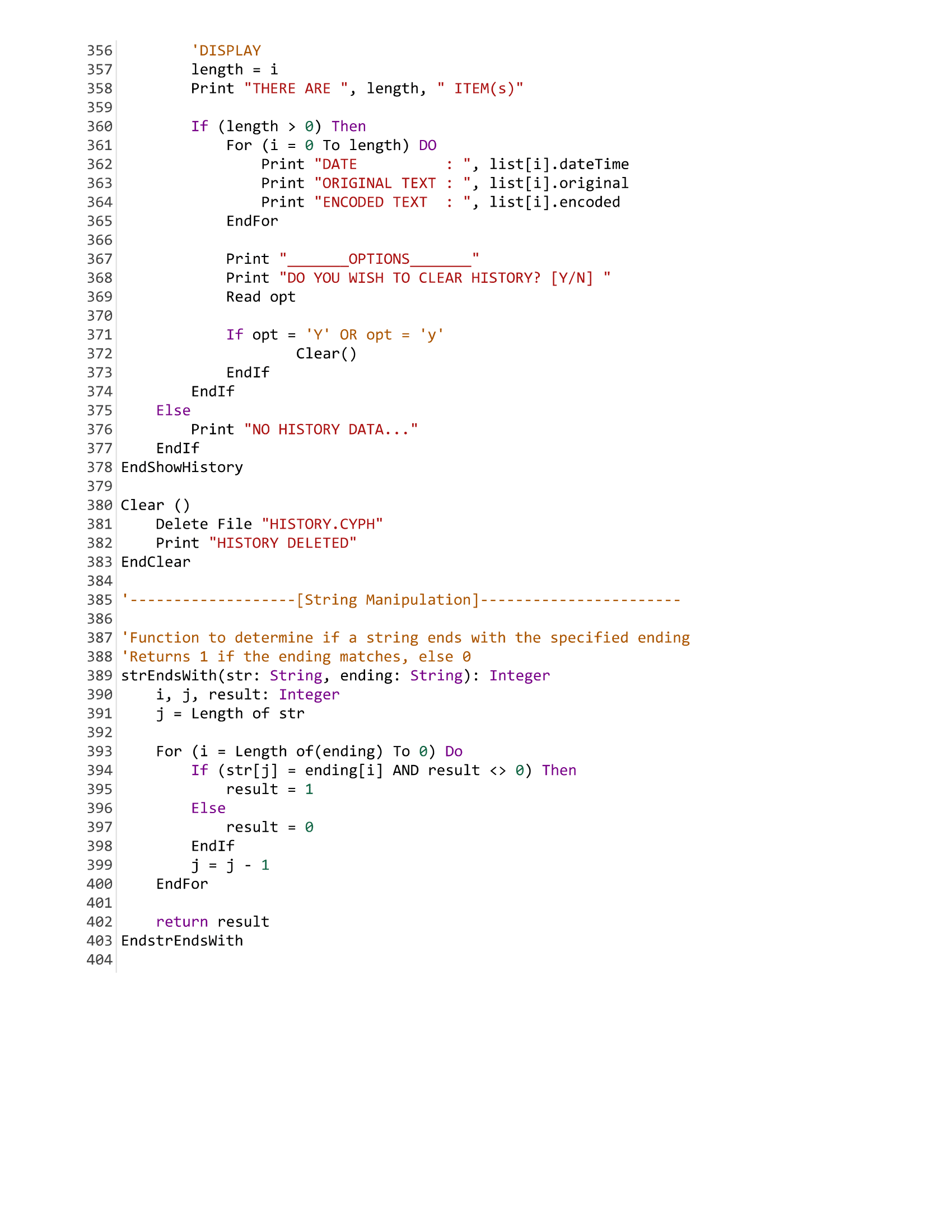
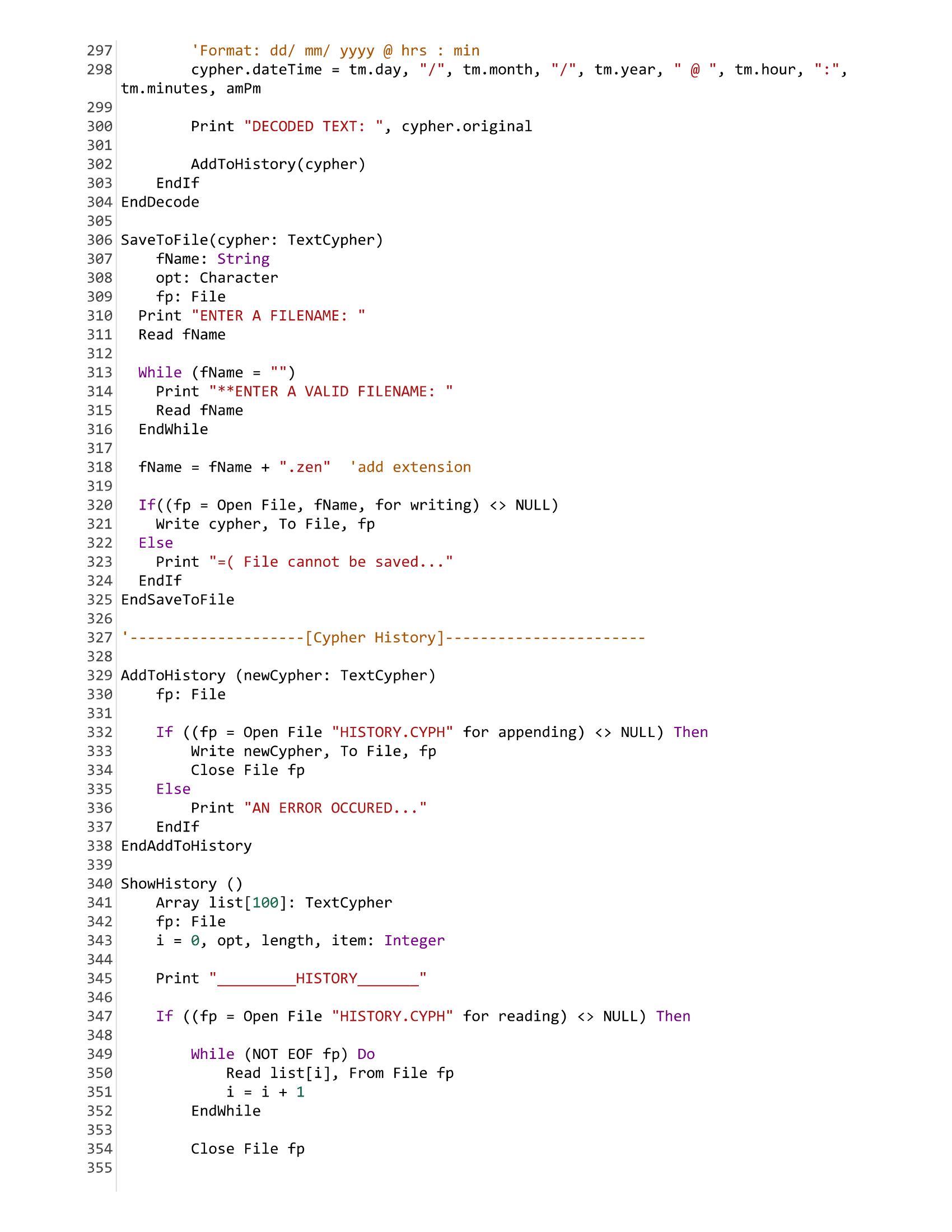
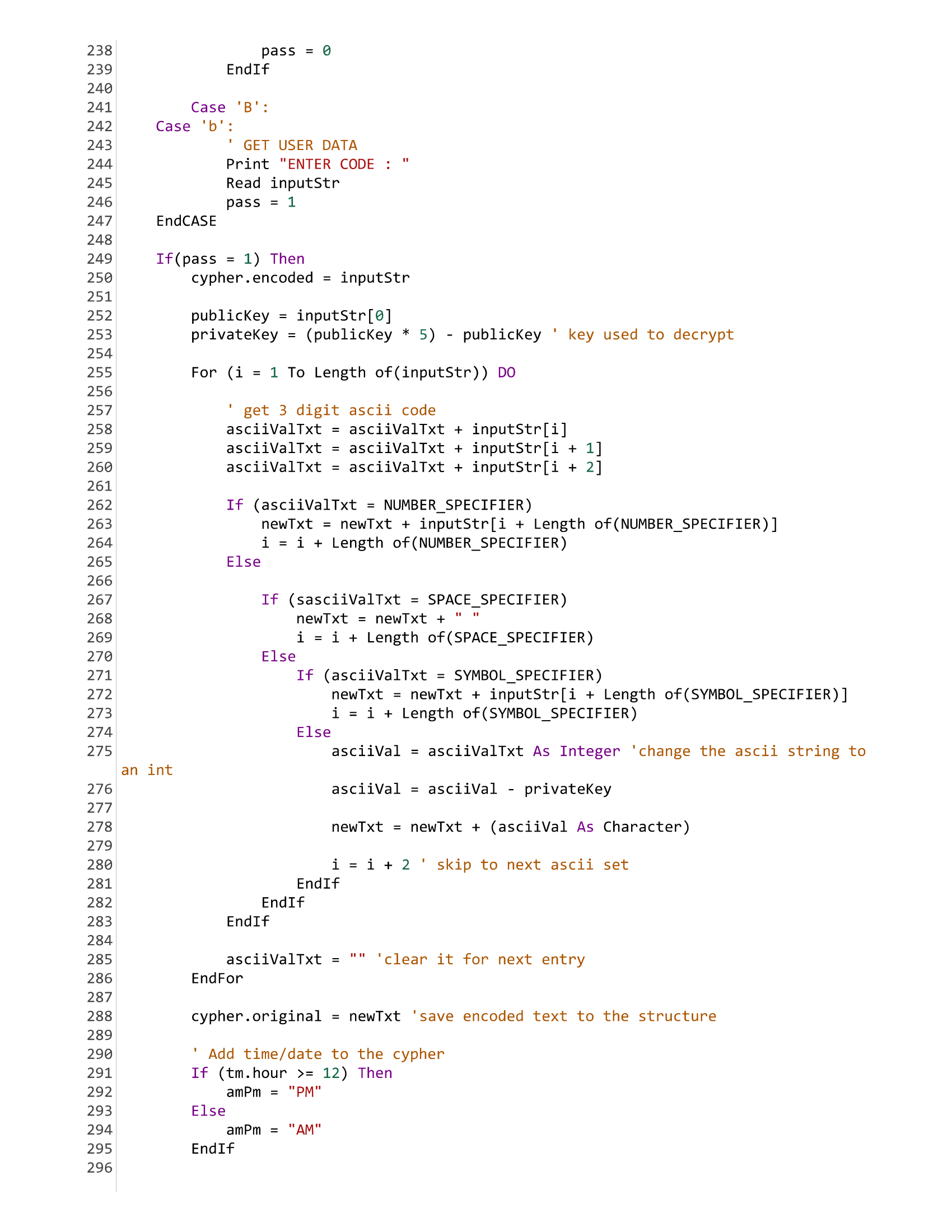
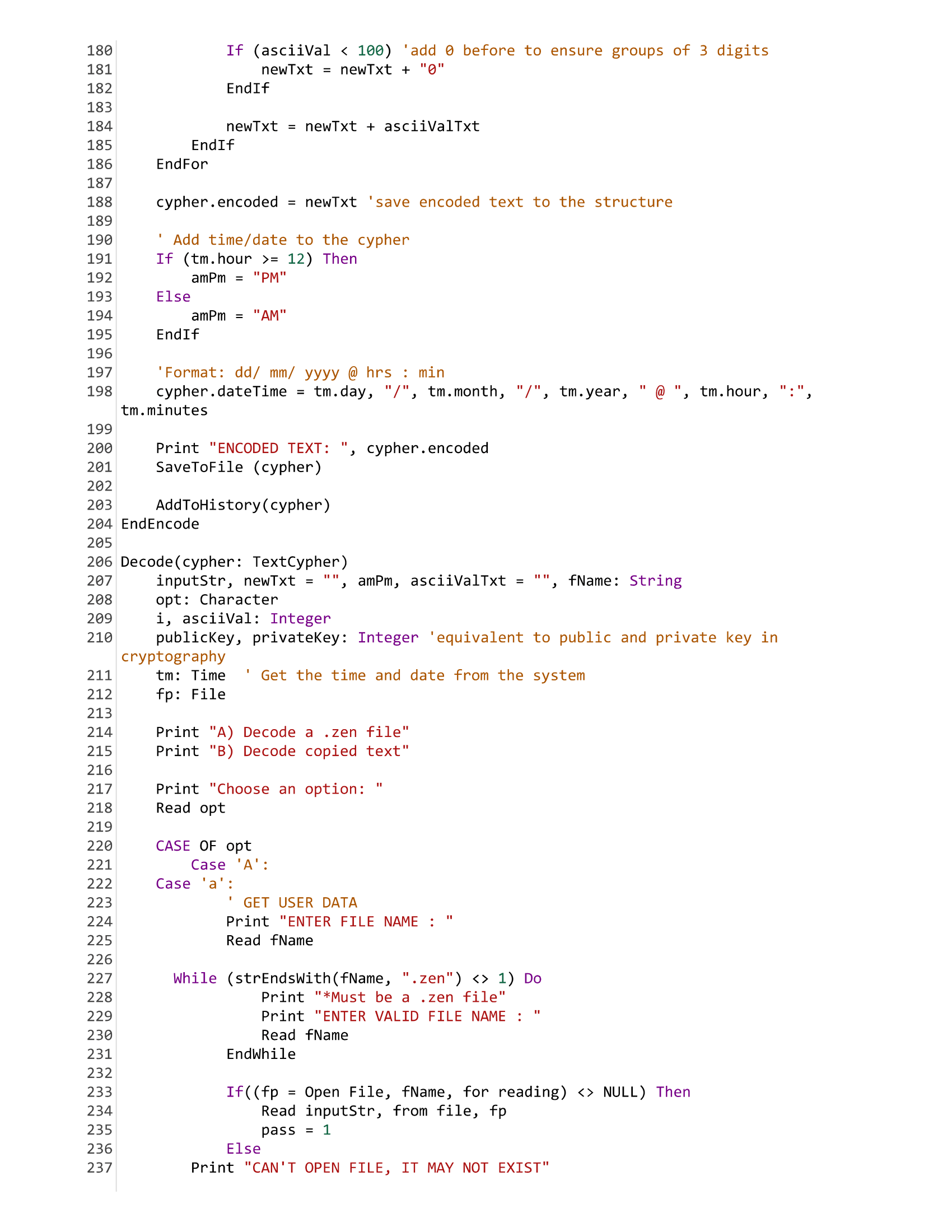
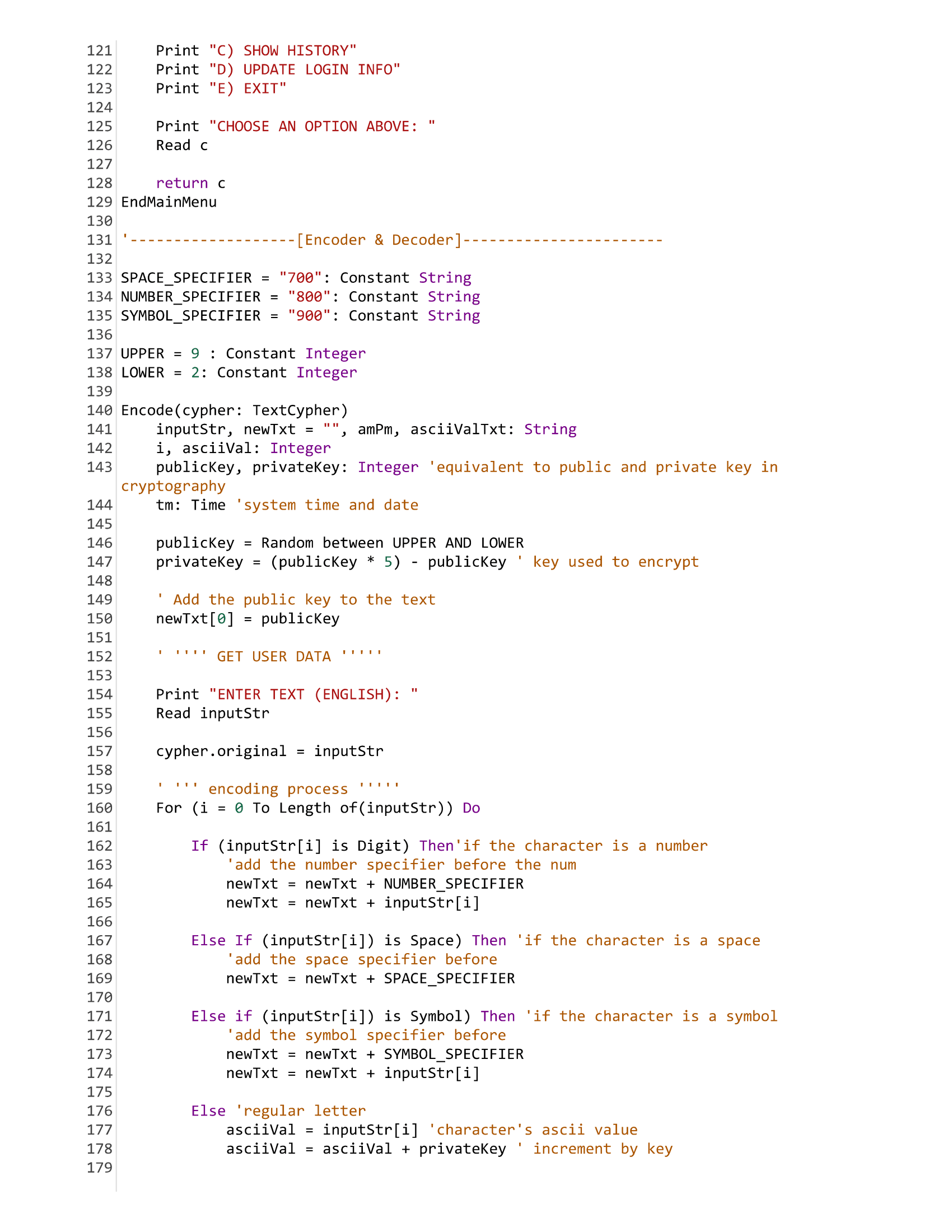
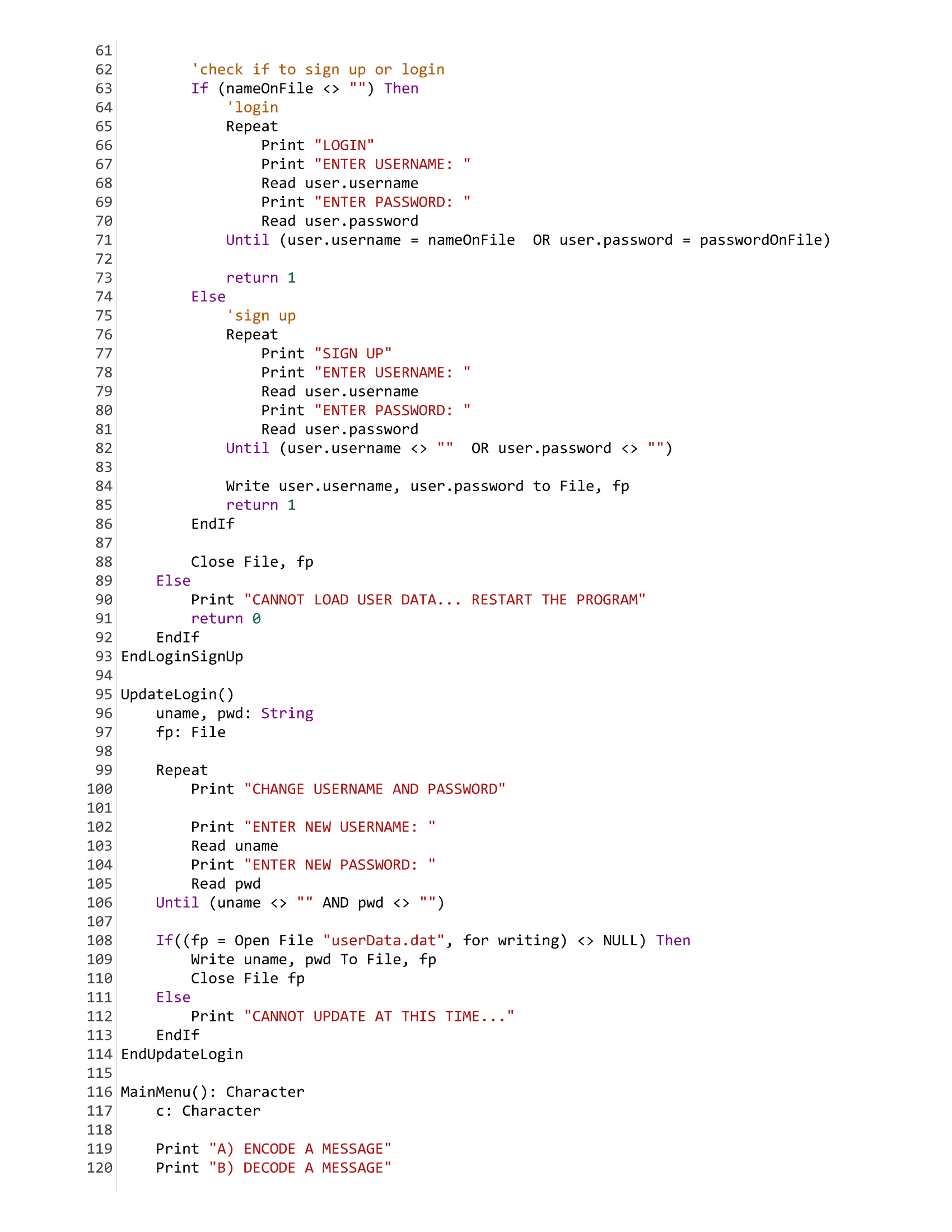
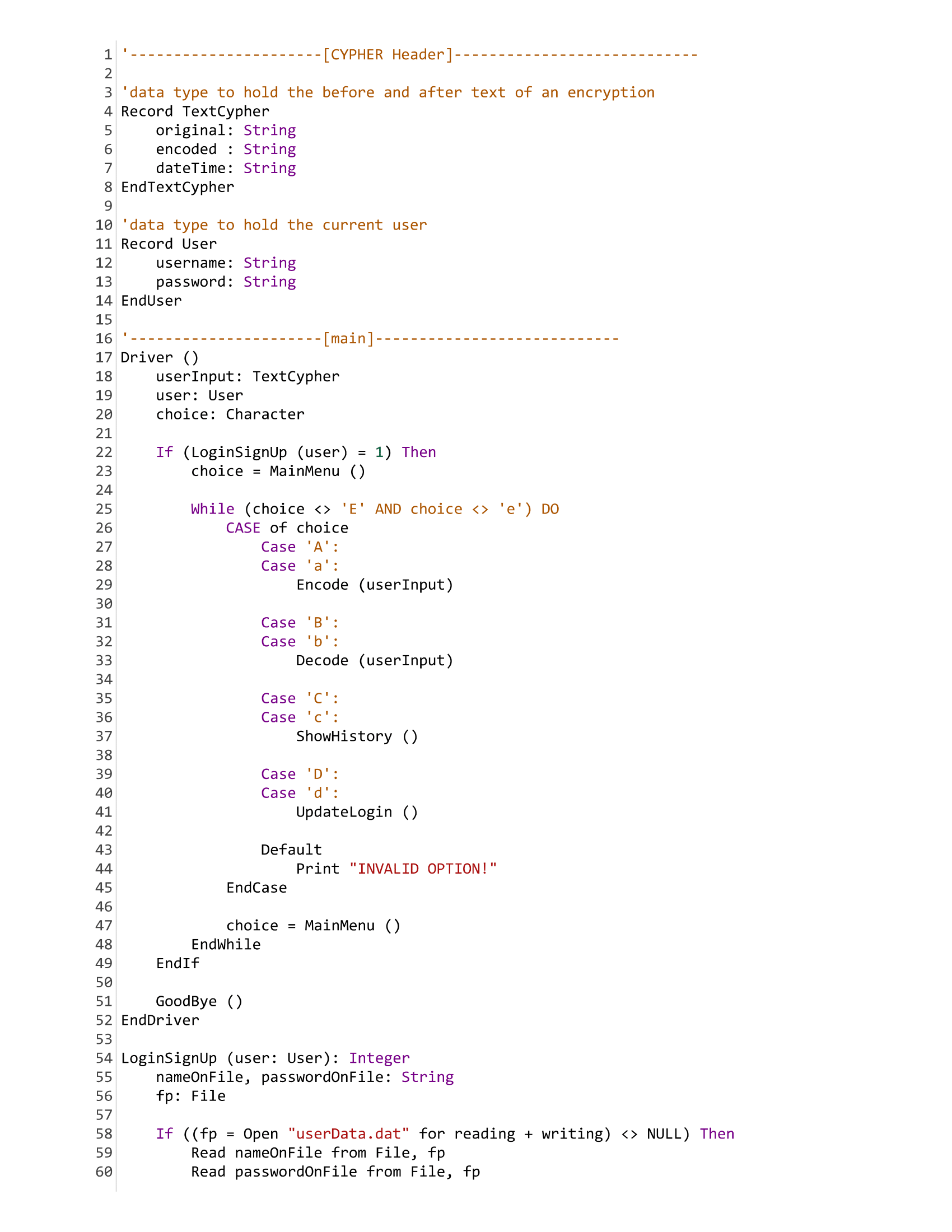
**SYSTEM REQUIREMENTS**

* The system shall require user sign in by username and password 
* The system shall accept user input as plain English and produce a cyphered version of the text as well as store it in a file 
* All cyphered text shall be saved in a separate file for later recollection
* The system shall allow users to input a custom filename for saving
* The system shall allow the user to decoder cyphered text and files

Non-Functional Requirements:

* The program should provide a simple, user friendly interface
* The program should allow more than one user accounts
* The system should allow the user to modify their login details at any point
* The system should allow users to reset password if forgot

**PSEUDOCODE**



**C CODE**

