New Jersey Institute of Technology — NES Encryptor Completion Team Assignment
CS490 — Summer 2024

Deliverables.

1) Completed NESEncryptor system running in a Visual Studio 2022 project.
2) Detailed Design Whitepaper documenting the design baseline.
3) In-Class demo (details to be provided)

Completed NESEncryptor system running under Visual Studio 2022:

I will provide the framework without the following classes/functions:

NESEncryptorControl
NESEncryptorFacade
WheelAssy.cpp (.h is provided)
main

main() should create the NESEncryptorControl object and then calls a start() function.

WheelAssy.cpp needs to implement the following:

```
advance()
    Set w2AtNotch to true if wheel2 is at notch
    If wheel 1 is at notch advance() wheel 2
    If w2AtNotch AND wheel 2 is NOT at notch advance() wheel 3
    advance() wheel 1
```

reset() resets the wheels to their initial positions

```
rToL(unsigned char c)
    Set c1 to Wheel 1 rToL(c)
    Set c2 to Wheel 2 rToL (c1)
    Set c3 to Wheel 3 rToL (c2)
    Return c3
```

```
lToR(unsigned char c)
    Set c3 to Wheel 3 lToR(c)
    Set c2 to Wheel 2 lToR(c3)
    Set c1 to Wheel 1 lToR(c2)
    Return c1
```

NESEncryptorControl

```
start()
        Set b to authorize() in ACSInterface
        If b is false, terminate with an error message
        Do forever
          Set action to getUserAction() in UserInput   // program terminates if user commands
          If ENC
            call encrypt() in NESEncryptorFacade
            call sendEncryptOK() in AASInterface
          If DEC call decrypt() in NESEncryptorFacade
            Call decrypt() in NESEncryptorFacade
            Call sendDecryptOK in AASInterface
```

NESEncryptorFacade

```
encrypt()
        Open red file for read in RedFileInterface
        Open black file for write in BlackFileInterface
        Reset wheels to initial positions

        While file not at end of file
              Set c to getNextChar() in the red file
              Set c to getPBC(c)
              Set c to rToL(c) in WheelAssy
              Set c to reflect(c) in Reflector
              Set c to lToR(c) in WheelAssy
              Set c to getPBC(c)
              Call advance() in WheelAssy
              Call putNextChar(c) in the black file
        Close red file
        Close black file

decrypt()
        Open black file for read in BlackFileInterface
        Open red file for write in RedFileInterface
         Reset wheels to initial positions
        While file not at end of file
              Set c to getNextChar() in the Red file
              Set c to getPBC(c)
              Set c to rToL(c) in WheelAssy
              Set c to reflect(c) in Reflector
              Set c to rToL(c) in WheelAssy
              Set c to getPBC(c)
              Call advance() in WheelAssy
              Call putNextChar(c) in the red file
        Close red file
        Close black file
```

Detailed Design Whitepaper documenting the design baseline

The Whitepaper should include:

1) a class diagram with all the operations, attributes and name of each class. Cardinality should be shown.
2) Sequence and collaboration diagrams tracing the thread of execution for the decrypt and encrypt operations.
3) Description of the subsystem
4) Description of each class
5) Traceability Matrix mapping the entity, boundary and control objects from the RAD into the implemented classes.

I have included below a partial class diagram created from Enterprise Architect along with the Enterprise Architect project. Your diagram should have all operations and attributes for each class shown. Enterprise Architect allow allows you to create sequence and collaboration (interaction) diagrams. You do not have to use Enterprise Architect but I would recommend it. A 30-day trial version is available from the Enterprise architect website.

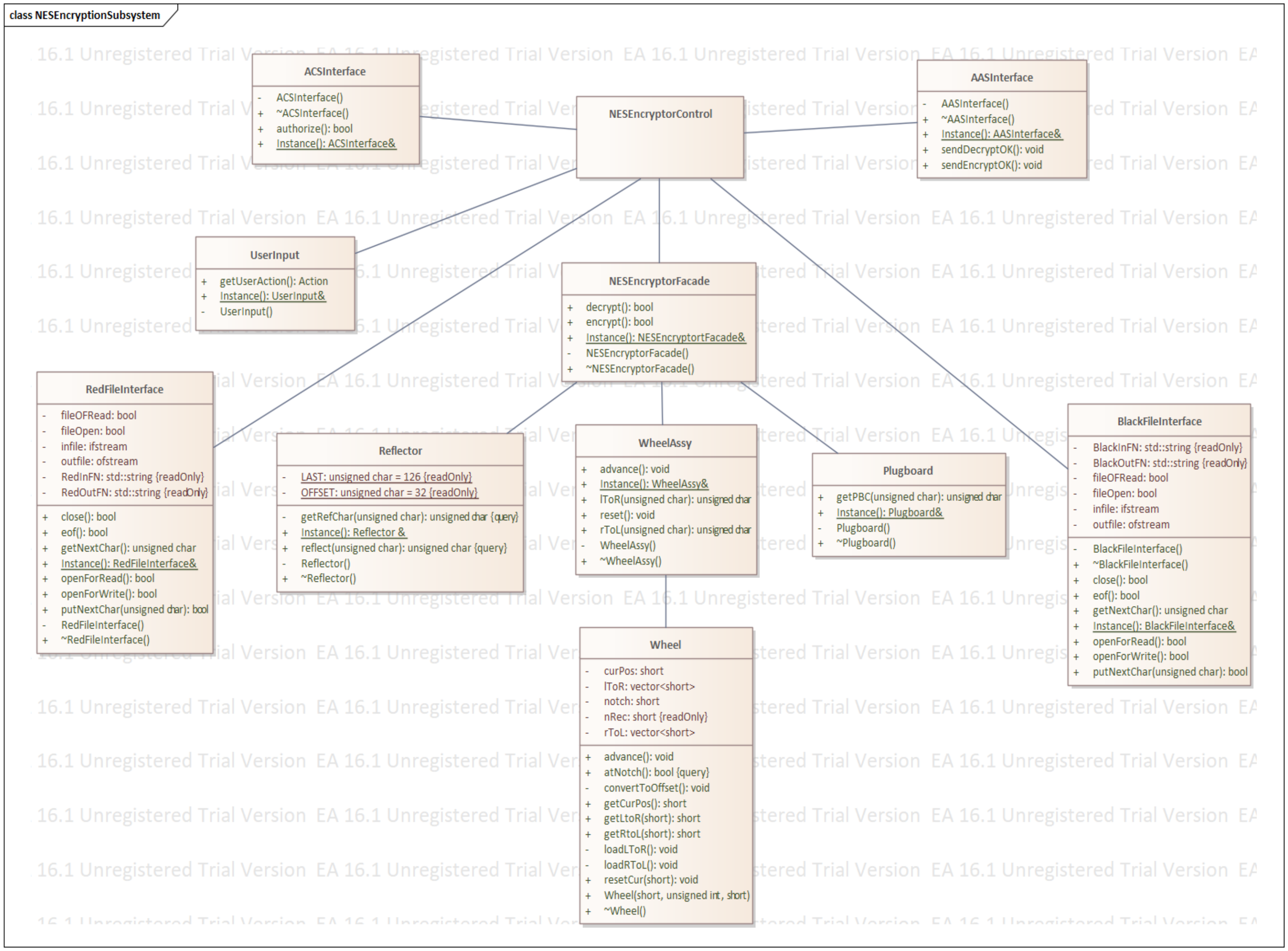
NOTE: You will need to “fix” the reflector to adjust for the 32 offset. The reflector needs to start at 0.

Submit a .zip file containing the Visual Studio project along with the detailed design whitepaper.

Grading:

Working program submitted	40%
Design Document described above	30% (6% for 1-5 above)
In-class demo	30%

The class diagram should look like this with all functions and attributes shown and correct associations and cardinalities:


class NESEncryptionSubsystem
ACSInterface
- ACSInterface()
+ ~ACSInterface()
+ authorize(): bool
+ Instance(): ACSInterface&
NESEncryptorControl
AASInterface
- AASInterface()
+ ~AASInterface()
+ Instance(): AASInterface&
+ sendDecryptOK(): void
+ sendEncryptOK(): void
UserInput
+ getUserAction(): Action
+ Instance(): UserInput&
- UserInput()
NESEncryptorFacade
+ decrypt(): bool
+ encrypt(): bool
+ Instance(): NESEncryptortFacade&
- NESEncryptorFacade()
+ ~NESEncryptorFacade()
RedFileInterface
- fileOFRead: bool
- fileOpen: bool
- infile: ifstream
- outfile: ofstream
- RedInFN: std::string {readOnly}
- RedOutFN: std::string {readOnly}
+ close(): bool
+ eof(): bool
+ getNextChar(): unsigned char
+ Instance(): RedFileInterface&
+ openForRead(): bool
+ openForWrite(): bool
+ putNextChar(unsigned char): bool
- RedFileInterface()
+ ~RedFileInterface()
Reflector
- LAST: unsigned char = 126 {readOnly}
- OFFSET: unsigned char = 32 {readOnly}
- getRefChar(unsigned char): unsigned char {query}
+ Instance(): Reflector &
+ reflect(unsigned char): unsigned char {query}
- Reflector()
+ ~Reflector()
WheelAssy
+ advance(): void
+ Instance(): WheelAssy&
+ lToR(unsigned char): unsigned char
+ reset(): void
+ rToL(unsigned char): unsigned char
- WheelAssy()
+ ~WheelAssy()
Plugboard
+ getPBC(unsigned char): unsigned char
+ Instance(): Plugboard&
- Plugboard()
+ ~Plugboard()
BlackFileInterface
- BlackInFN: std::string {readOnly}
- BlackOutFN: std::string {readOnly}
- fileOFRead: bool
- fileOpen: bool
- infile: ifstream
- outfile: ofstream
- BlackFileInterface()
+ ~BlackFileInterface()
+ close(): bool
+ eof(): bool
+ getNextChar(): unsigned char
+ Instance(): BlackFileInterface&
+ openForRead(): bool
+ openForWrite(): bool
+ putNextChar(unsigned char): bool
Wheel
- curPos: short
- lToR: vector<short>
- notch: short
- nRec: short {readOnly}
- rToL: vector<short>
+ advance(): void
+ atNotch(): bool {query}
- convertToOffset(): void
+ getCurPos(): short
+ getLtoR(short): short
+ getRtoL(short): short
- loadLToR(): void
- loadRToL(): void
+ resetCur(short): void
+ Wheel(short, unsigned int, short)
+ ~Wheel()