

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.

Здесь и далее в качестве шифра используется AES-128 в режиме CTR. В качестве кода аутентичности HMAC-SHA-256.

Ожидаемый формат шифртекста: `CTR(TV) || ENCRYPTED_DATA || MAC`, где `MAC` вычисляется от `(CTR(TV) || ENCRYPTED_DATA)` на ключе, независимом от ключа шифрования.

1. Используя средства языка (реализации AES-CTR и HMAC-SHA-256) реализовать класс, реализующий аутентифицированное шифрование в режиме Encrypt-Then-Mac, со следующим интерфейсом:

1.0 Конструктор `AuthenticEncryptor(Mode mode)`

- инициализирует объект для шифрования или расшифрования, определяемый значением переменной `mode`.

1.1 `void setKey(byte[] key)`

- инициализирует объект шифрования ключом `key`.

1.2 `byte[] AddBlock(byte[] dataBlock, bool isFinal)`

- добавляет блок данных для аутентифицированного зашифрования или расшифрования. В случае передачи флага `isFinal` должен вычисляться\проверяться результирующий код аутентичности (в зависимости от режима `mode`). При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка. В случае зашифрования данная функция зашифровывает блок данных, и добавляет блок шифртекста для вычисления кода аутентичности, если блок последний - дополняет возвращаемый шифртекст результирующим кодом аутентичности. В случае расшифрования добавляет шифртекст для вычисления кода аутентичности, расшифровывает его, если блок последний - проверяет код аутентичности, содержащийся в блоке с полученным, на основе переданного шифртекста.

1.3 `byte[] ProcessData(byte[] data)`

Производит шифрование\расшифрование переданных данных через вызовы `AddBlock`. При расшифровании, в случае неуспешной проверки кода аутентичности должна выводиться ошибка.

2. Зашифровать и проверить произвольный блок данных, размера 100 МВ.