```
// please run this command on linux before running code
 //sudo apt-get install libcrypto++-utils libcrypto++8 libcrypto++-dev
libcrypto++-doc
```

Summary:

The `CryptoApp` class provides a graphical user interface (GUI) for encrypting and decrypting text using AES and Blowfish algorithms. It also allows for file encryption and decryption using these algorithms. The app prompts the user to enter an encryption key, then performs the encryption or decryption based on user input.

Class `CryptoApp`:

cpp

```
class CryptoApp : public QWidget {
```

- Inherits from `QWidget`, making it a widget that can be shown as a window.

Public Methods:

cpp

```
public:
    CryptoApp(QWidget *parent = 0);
```

- Constructor for the `CryptoApp` class. Sets up the GUI elements including input/output text areas, buttons for encryption/decryption, and file encryption/decryption.

Private Slots:

cpp

```
private slots:
    void fileEncryptAes();
    void fileDecryptAes();
    void fileEncryptBlowfish();
    void fileDecryptBlowfish();
    void encryptAes();
    void decryptAes();
    void encryptBlowfish();
    void decryptBlowfish();
    void refreshApp();
    string getEncryptionKeyFromUser();
```

- These are private slots (functions) that handle various actions triggered by button clicks.
- `fileEncryptAes()`: Encrypts a selected file using AES.
- `fileDecryptAes()`: Decrypts a selected file using AES.
- `fileEncryptBlowfish()`: Encrypts a selected file using Blowfish.
- `fileDecryptBlowfish()`: Decrypts a selected file using Blowfish.
- `encryptAes()`: Encrypts input text using AES.
- `decryptAes()`: Decrypts input text using AES.
- `encryptBlowfish()`: Encrypts input text using Blowfish.
- `decryptBlowfish()`: Decrypts input text using Blowfish.
- `refreshApp()`: Resets the app, clearing keys and texts.
- `getEncryptionKeyFromUser()`: Prompts the user for an encryption key and returns it as a string.

Private Members:

cpp

```
private:
    QTextEdit *inputText;
    QTextEdit *outputText;
    byte iv_aes[AES::BLOCKSIZE];
    byte iv_blowfish[Blowfish::BLOCKSIZE];
    string encryptedText_aes, encryptedText_blowfish;
```

- `inputText`: A QTextEdit widget for user input.
- `outputText`: A QTextEdit widget for displaying output.
- `iv_aes`: Initialization vector for AES algorithm.
- `iv_blowfish`: Initialization vector for Blowfish algorithm.
- `encryptedText_aes`: Stores encrypted text using AES.
- `encryptedText_blowfish`: Stores encrypted text using Blowfish.

Main Function:

cpp

```
int main(int argc, char *argv[]) {
    QApplication app(argc, argv);

    CryptoApp cryptoApp;
    cryptoApp.show();

    return app.exec();
}
```

- The `main` function creates an instance of `QApplication` and `CryptoApp`.
- Shows the `CryptoApp` GUI and starts the event loop.

Note:

- The class uses Crypto++ library for AES and Blowfish encryption/decryption.
- Each encryption/decryption function prompts the user for an encryption key.
- File encryption/decryption functions allow the user to select input and output files via a file dialog.
- The GUI includes buttons for various encryption/decryption operations.
- Output is displayed in the `outputText` QTextEdit widget.

Important Points:

- The code includes error handling using `try-catch` blocks for Crypto++ exceptions.
- Input and output text fields are updated with the results of encryption/decryption operations.
- Initialization vectors (`iv_aes` and `iv_blowfish`) are randomly generated for each encryption operation.
- The `refreshApp()` function clears sensitive data and resets the app for further use.