1. readyRead()信号在有数据到来时发出
2. disconnected()信号在断开连接时发出
3.bytesAvailable() Returns the number of incoming bytes that are waiting to be read.
4.qint64 QIODevice::read(char *data, qint64 maxSize)
        Reads at most maxSize bytes from the device into data, and returns the number of bytes read. If an error occurs, such as when attempting to read from a device opened in WriteOnly mode, this function returns -1.
        0 is returned when no more data is available for reading. However, reading past the end of the stream is considered an error, so this function returns -1 in those cases (that is, reading on a closed socket or after a process has died).

5.QAbstractSocket::setSocketOption(QAbstractSocket::SocketOption option, const QVariant &value)
Sets the given option to the value described by value.

6.
qint64 QIODevice::write(const char *data, qint64 maxSize)
Writes at most maxSize bytes of data from data to the device. Returns the number of bytes that were actually written, or -1 if an error occurred.

7.QString QString::left(int n) const
Returns a substring that contains the n leftmost characters of the string.
The entire string is returned if n is greater than or equal to size(), or less than zero.

```
  QString x = "Pineapple";
  QString y = x.left(4);      // y == "Pine"
```

/home/yin/asdf

```
void MainWindow::on_pushButton_clicked()
{

        QFile file( filename );

        if( !file.open( QIODevice::ReadOnly | QIODevice::Text ) )
            QMessageBox::warning(this, tr("Error opening!"), tr
("Could not open the file"));

        QTextStream stream( &file );

        while( !stream.atEnd() )
        {
            QString lineText;
```

```
            lineText = stream.readLine(); //Read a line of text

            QStringList tokens= lineText.split("
",QString::SkipEmptyParts); //Take tokens from the line
        }

        file.close();
}
```