# سریال بخش سوم

در این بخش هدف ایجاد یک ارتباط سریالی کامل می‌باشد که می‌توان از آن برای ارسال اطلاعات بین برنامه‌های مختلف و یا سخت افزار های مختلف استفاده کرد. به این منظور ما باید با فرمت جدیدی از اطلاعات به نام استرینگ (زنجیره‌ای از کاراکتر) و ابزار هایی که برای ایجاد تغییرات در این نوع داده نیاز است آشنا شویم.

# خواندن اطلاعات به صورت زنجیره‌ای از داده ها

در قسمت سریال قسمت دوم گفته شد که چگونه اطلاعات ارسال شده از طریق پورت سریال را به صورت یک کاراکتر در لحظه بخوانیم. مشکل این نحوه‌ای خواندن سخت بودن آن می‌باشد. فرض کنید دستگاه شما می‎‌خواهد بعد از گرفتن کلمه "شروع" شروع به خواندن دما کند. برای این کار شما باید به صورت کاراکتر به کاراکتر ورودی های پورت سریال را بررسی کنید. روش دیگر این است که داده ها را به صورت بسته های بزرگتر دریافت کنیم و همین طور به صورت بسته های بزرگتر بفرستیم. به این بسته ها که مجموعه‌ای از کارکتر ها هستند، استرینگ یا رشته گفته می‌شود.

## String یا رشته

مجموعه‌ای از کاراکتر های وابسته به هم را رشته می‌گویند. این مجموعه می‌تواند در بر گیرنده هر نوع کاراکتری باشد. مانند آرایه‌ای از کاراکتر ها می‌باشد.



توجه: همان طور که گفته شد رشته را می‌توان به صورت آرایه‌ای از کاراکتر ها نیز تعریف کرد. تفاوت این روش با روش قبلی این است که در روش قبلی نوع داده رشته بوده ولی در این روش نوع داده آرایه‌ای از کاراکتر ها می‌باشد.

توجه2: لازم به ذکر است که برای برابر قرار دادن یک متغیر و یک کاراکتر حتما باید کاراکتر را در ‘’ تعریف کنید.



# ارتباط سریالی با مفهومی گسترده تر

در این بخش می‌خواهیم از طریق پورت سریال و استفاده از رشته‌ای از داده ها، ارتباطی گسترده تر ایجاد کنیم. در این بخش می‌خواهیم با ارسال “On&1” دیود نوری شماره 1 را روشن کنیم. با ارسال “On&2” دیود نوری شماره 2 را روشن کنیم و با ارسال “Off” تمامی دیود ها را خاموش کنیم. برای این کار ابتدا با توابع زیر آشنا بشویم.

## Serial.readString()

این تابع اطلاعات داخل بافر را به صورت یک رشته از کاراکتر ها می‌خواند تا زمانی که زمانش تمام شود. به صورت پیش فرض 1 ثانیه صبر می‌کند و تمامی اطلاعات فرستاده شده در بافر را به صورت یک رشته از کاراکتر ها دریافت می‌کند.



## Serial.flush()

این تابع تمامی اطلاعات داخل بافر را تخلیه می‌کند. در واقع به برنامه می‌گوید که تمامی اطلاعات داخل بافر را بخواند و بافر را تخلیه کند. معمولا بعد از تابع Serial.readString برای اینکه از خوانده شدن تمامی اطلاعاتی که در یک پکیج بوده‌اند مطمئن شویم از این تابع استفاده می‌کنیم.



## String.substring()

این یکی از توابع مربوط به کلاس رشته می‌باشد. این تابع به عنوان ورودی 2 درایه می‌گیرد و به وسیله آن رشته را قطعه می‌کند. به بیان دیگر کاراکتر های بین آن 2 درایه را به صورت یک رشته دیگر خروجی می‌دهد.



توجه: می‌توان به این تابع یک ورودی نیز داد. در این صورت این تابع رشته را از آن درایه تا به انتها جدا خواهد کرد.

## String.length()

این تابع طول آرایه رشته را خروجی می‌دهد.



## String.toInt()

این تابع یک رشته از کارکترها را به یک عدد صحیح تبدیل می‌کند.



## ابزار های کاربردی برای استفاده از رشته ها

تعداد بسیار زیادی از این ابزارک ها وجود دارند که ما بخشی از آنها را در اینجا مطرح کردیم. در این [لینک](https://www.arduino.cc/en/Tutorial/BuiltInExamples) در بخش مربوط به رشته ها، شما می‌توانید تمامی این ابزار های از پیش تعیین شده را پیدا کنید. همین طور شما می‌توانید با استفاده از مثال های پیش فرض آردینو که مربوط به این بخش می‌باشند، با این ابزارک ها آشنا شوید.

# استاندار های ارسال متن

برای نمایش یک متن در کامپیوتر، شما علاوه بر محتویات متن نیازمند محل قرار گیری متن نیز می‌باشید. به طور مثال اگر شما بخواهید در خط اول بنویسید "سلام" و بعد به خط بعدی بروید و بنویسید "خداحافظ" باید به نحوی به کامپیوتر خود بفهمانید که به خط بعد برود. به همین دلیل نماد هایی برای این چنین عملیات هایی استفاده می‌شود.

## \n یا خط بعد

این علامت اگر از طریق سریال ارسال شود و توسط کامپیوتر دریافت شود به این معنی می‌باشد که نشانگر کامیوتر به خط بعدی منتقل شود. توجه کنید که به معنی اول خط بعد نمی‌باشد و فقط به خط بعد می‌رود.



## \r یا ابتدای خط

این علامت به این معناست که نشانگر به ابتدای خطی که در آن قرار دارد حرکت کند.



## \t یا فاصله افقی

این علامت نشانگر را به اندازه یک tab در راستای افقی جابجا می‌کند.



## \n\r یا ابتدای خط بعد

این علامت تلفیقی از 2 علامت قبلی است و نشانگر را به ابتدای خط بعدی منتقل می‌کند.



نکته: در ابتدا شاید این علامت ها نیاز نباشند اما مشکل اصلی نیازمندی شما به این علامت ها نیست بلکه شما قطعا برای برقراری ارتباط با نرم افزار های دیگر با این علامت ها مواجه می‌شوید و باید حضور این علامت ها را در رشته کاراکتری خود در نظر بگیرید و بر این اساس روی رشته ها عملیات های پردازشی انجام دهید.

# برقراری ارتباط سریالی و پردازش روی رشته کاراکترها

همان طور که گفته شد هدف از این بخش پردازش روی رشته‌ای از کارکتر ها می‌باشد. نوع رشته های ارسالی به سبک "شماره دیود نوری& وضعیت اعمالی به آن" می‌باشد مانند "On&1". این به این معناست که دیود شماره یک را روشن کنید. در عین حال پیام "Off&1" به این معناست که دیود 1 را خاموش کنید. همین طور پیام "Blink&1%3" یعنی دیود 1 را برای مدت فقط 3 ثانیه روشن شود و بعد خاموش شود. ما می‌خواهیم با ارسال این پیام ها از طریق پورت سریال آردینو، دیود ها را خاموش و روشن کنیم. برای این کار باید بر روی رشته هایی که از طریق پورت دریافت می‌کنیم پردازش انجام دهیم. ابتدا رشته را به یک زیر رشته تبدیل می‌کنیم که از ابتدا تا اولین & را در بر بگیرد. سپس این زیر رشته را بررسی می‌کنیم و نوع دستور را بدست می‌آوریم. سپس با توجه به اینکه نوع دستور چیست تابع مورد نظر آن را اجرا می‌کنیم. اگر نوع دستور روشن یا خاموش بود اولین کاراکتر بعد از & را به عنوان شماره دیود در نظر می‌گیریم و عملیات به اتمام می‌رسد. اگر دستور چشمک زن ارسال شده بود، اولین کاراکتر بعد از & را به عنوان شماره دیود نوری و اولین کارکتر بعد از % را به عنوان زمان روشن ماندن دیود در نظر می‌گیریم. حال اگر تعداد دیود ها و یا زمان روشن ماندن قرار بود اعدادی بین 1 تا 1000 باشند به جای بررسی کردن فقط اولین کاراکتر بعد از & تمامی کاراکتر های بین & و % را در نظر می‌گیریم.



برای تنوع و بهتر فهمیدن موضوع، نمونه‌ی دیگری اما ساده از این گونه نرم افزار ها را در ادامه بررسی می‌کنیم. شما ابتدا به این نرم افزار دو کاراکتر مشخص می‌دهید. سپس یک رشته به آن می‌دهید که این دو کاراکتر در آن استفاده شده باشد. نرم افزار به شما تمامی کاراکتر های بین این دو کاراکتر مشخص را بر می‌گرداند. از این نرم افزار می‌توانید به عنوان یک پایه برای نوشتن برنامه های بسیار پیچیده تر استفاده کنید.

