1. **split** — useful for splitting tab-separated or colon-separated data, etc.
   
   - You can use regular expressions to split the string, but the default is to split on whitespace

   ```perl
   $input_data = "sandra:denise:theo:vanessa:rudy";
   @my_list = split(/:/, $input_data);

   while (<>) {
     @my_list = split; # equivalent to: @my_list = split(/\s+/, $_);
   }

   - To get each individual character, you can split on nothing

   ```perl
   @new_list = split//, $input_data;
   foreach (@new_list) {
     print "$_\n";
   }
   ```

2. **index** — find a substring
   
   - Format: `index(string, substring)`
   - Only returns the location of the *first* match

   ```perl
   $long = "this is a great discussion";
   $location = index($long, "great"); # 10

   - Has an optional parameter which allows you specify the position to start searching—can loop over this
   - When it can’t find anything which matches, `index` returns -1

   ```perl
   $where = 0;
   while ($where != -1) {
     $where = index($long, "is", $where+1);
     print "Locations are: $where\n"; # 2, 5, 17, -1
   }
   ```
3. `substr` — manipulate a substring

- Allows you to grab a substring of a particular string, if you know the beginning and end of it
- Format: `substr(string, start, length)`

```php
$middle = substr($long, 5, 8);
print "$middle
"; # "is a gre"
```

- Can replace the substring, too (even if a different length)
  - Later, we’ll see how to use, s/// (or s///g) to replace strings

```php
# replace a substring
substr($long, 5, 2) = "used to be";
print "$long
"; # "this used to be a great discussion"
```