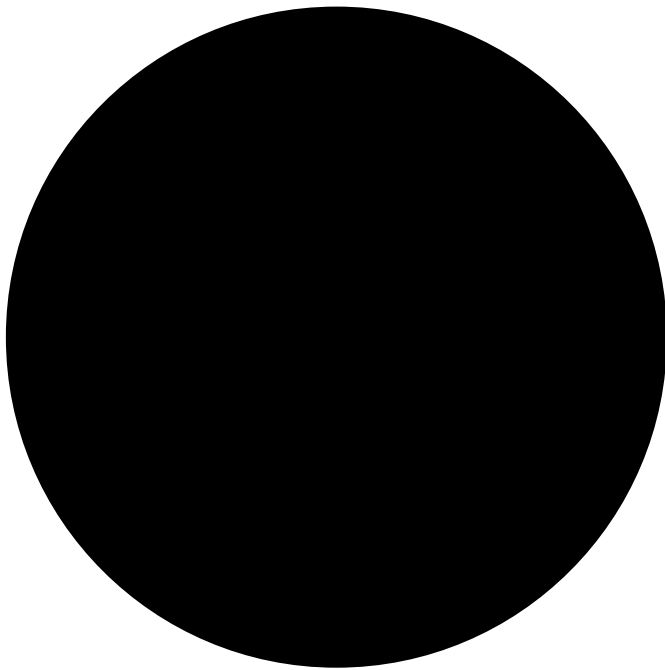


# Eli's Wolfram Language Cheat Sheet

```
In[*]:= (*@ puts brackets around a thing*)  
f@{g, h, i}
```

```
Out[*]=  
f[{g, h, i}]  
  
(*it is useful for functions that only require one input*)
```

```
In[*]:= Graphics@Disk[]  
Out[*]=
```



```
In[*]:= Sqrt@4
```

```
Out[*]=  
2
```

```
In[*]:= IntegerQ@10
```

```
Out[*]=  
True
```

```
(*@@ replaces list brackets and inserts  
the whole list directly into the function*)  
f@@{g, h, i}
```

```
Out[*]=  
f[g, h, i]
```

(\*this works for functions that take more than one argument\*)

```
In[ ]:= NestList@@{#^2&, 2, 2}
```

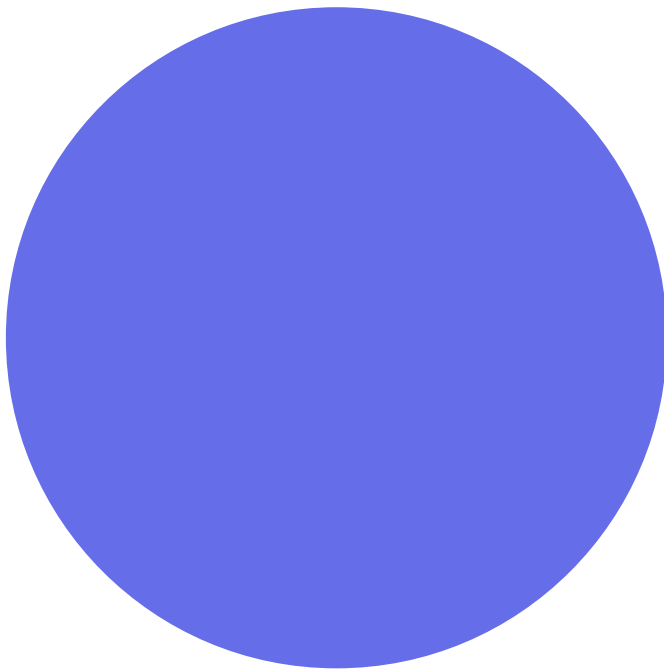
```
Out[ ]:=  
{2, 4, 16}
```

```
In[ ]:= Table@@{x+y, {x, 10}, {y, 10}}
```

```
Out[ ]:=  
{ {2, 3, 4, 5, 6, 7, 8, 9, 10, 11}, {3, 4, 5, 6, 7, 8, 9, 10, 11, 12},  
  {4, 5, 6, 7, 8, 9, 10, 11, 12, 13}, {5, 6, 7, 8, 9, 10, 11, 12, 13, 14},  
  {6, 7, 8, 9, 10, 11, 12, 13, 14, 15}, {7, 8, 9, 10, 11, 12, 13, 14, 15, 16},  
  {8, 9, 10, 11, 12, 13, 14, 15, 16, 17}, {9, 10, 11, 12, 13, 14, 15, 16, 17, 18},  
  {10, 11, 12, 13, 14, 15, 16, 17, 18, 19}, {11, 12, 13, 14, 15, 16, 17, 18, 19, 20}}
```

```
In[ ]:= Graphics[Style@@{Disk[], RandomColor[]}]
```

```
Out[ ]:=
```



(\*@@@ operates on lists within a list\*)

```
In[ ]:= f@@@{{g}, {h}, {i}}
```

```
Out[ ]:=  
{f[g], f[h], f[i]}
```

```
In[ ]:= Plus@@@{{1, 2}, {3, 4}}
```

```
Out[ ]:=  
{3, 7}
```

```
In[ ]:= FromLetterNumber@@{{2}, {3}}
```

```
Out[ ]:=  
FromLetterNumber[{2}, {3}]
```

```

In[ ]:= FromLetterNumber@@@ {{2}, {3}}
Out[ ]:=
{b, c}

(*/@ does the same thing as @@@*)

In[ ]:= f /@ {g, h, i}
Out[ ]:=
{f[g], f[h], f[i]}

(*except /@ also works with pure functions*)

In[ ]:= #^2 & /@ {1, 2, 3}
Out[ ]:=
{1, 4, 9}

(*@@@ does not work with pure functions*)

In[ ]:= #^2 &@@@ {1, 2, 3}
Out[ ]:=
{1, 2, 3}

(*module is pretty simple. you set a condition
and then evaluate an equation with that condition*)

In[ ]:= Module[{x = 2}, 3 x + 5 x + 7]
Out[ ]:=
23

(*Module is functionally the same *)

(*Array works with a list of lists of the same size*)

In[ ]:= Array[f, 9, 2]
Out[ ]:=
{f[2], f[3], f[4], f[5], f[6], f[7], f[8], f[9], f[10]}

(*You know how to do table*)

In[ ]:= Table[x + y, {x, 3}, {y, 3}]
Out[ ]:=
{{2, 3, 4}, {3, 4, 5}, {4, 5, 6}}

(*/. replaces things from a list*)

In[ ]:= {a, b, c} /. c -> d
Out[ ]:=
{a, b, d}

(*an array is a list of lists*)

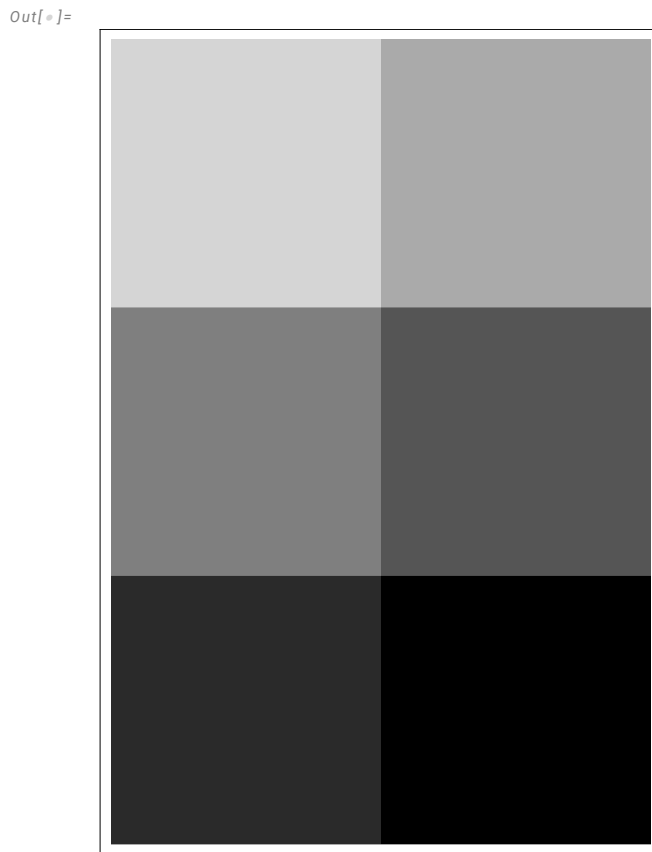
In[ ]:= ArrayQ[{1, 2}, {3, 4}, {5, 6}]
Out[ ]:=
False

```

```
In[ ]:= ArrayQ[{{1, 2}, {3, 4}, {5, 6}}]
```

```
Out[ ]:=  
True
```

```
In[ ]:= ArrayPlot[{{1, 2}, {3, 4}, {5, 6}}]
```



(\*You can get to an array by partitioning a list\*)

```
In[ ]:= ArrayQ[Partition[{1, 2, 3, 4}, 2, 1]]
```

```
Out[ ]:=  
True
```

(\*You can group stuff from arrays with transpose\*)

```
In[ ]:= Transpose[{{1, 2}, {3, 4}, {5, 6}}]
```

```
Out[ ]:=  
{{1, 3, 5}, {2, 4, 6}}
```

(\*You can also pull all of the first, second, etc. variables out of arrays\*)

```
In[ ]:= {{1, 2}, {3, 4}, {5, 6}}[[All, 1]]
```

```
Out[ ]:=  
{1, 3, 5}
```

```
In[ ]:= {{a, b}, {c, d}, {e, f}}[[2, 2]]
Out[ ]:=
d
```