Learning Wolfram Language

Exercises for Section 1 | Starting Out: Elementary Arithmetic

1.1 Compute1+2+3.

```
In[13]:=  1 + 2 + 3

Out[13]=  6
```

1.2 Add the numbers 1,2,3,4,5.

```
In[15]:=  1 + 2 + 3 + 4 + 5

Out[15]=  15
```

1.3 Multiply the numbers 1, 2, 3, 4, 5.

```
In[16]:=  1 * 2 * 3 * 4 * 5

Out[16]=  120
```

1.4 Compute 5 squared (i . e . 5*5 or 5 raised to the power 2)

```
In[17]:=  5 ^ 2

Out[17]=  25
```

1.5 Compute 3 raised to the fourth power .

```
In[18]:=  3 ^ 4

Out[18]=  81
```

1.6 Compute 10 raised to the power 12 (a trillion) .

```
In[20]:=  10 ^ 12

Out[20]=  1 000 000 000 000
```

1.7 Compute 3 raised to the power 7 x8 .

```
In[19]:=  3 ^ (7 * 8)

Out[19]=  523 347 633 027 360 537 213 511 521
```

1.8 Add parentheses to 4 - 2*3 + 4 to make 14.

```
In[21]:=  (4 - 2) * (3 + 4)
Out[21]=  14
```

1.9 Compute twenty - nine thousand mutiplied by seventy - three .

```
In[22]:=  29 000 * 73
Out[22]=  2 117 000
```

+1.1 Add all integers from - 3 to + 3.

```
In[23]:=  -3 + -2 + -1 + 1 + 2 + 3
Out[23]=  0
```

+1.2 Compute 24 divided by 3.

```
In[24]:=  24 / 3
Out[24]=  8
```

+1.3 Compute 5 raised to the power 100.

```
In[25]:=  5 ^ 100
Out[25]=  7 888 609 052 210 118 054 117 285 652 827 862 296 732 064 351 090 230 047 702 789 306 640 625
```

+1.4 Subtract 5 squared from 100

```
In[26]:=  100 - (5 ^ 2)
Out[26]=  75
```

+1.5 Multiply 6 by 5 squared, and add 7

```
In[27]:=  (6 * (5 ^ 2)) + 7
Out[27]=  157
```

+1.6 Compute 3 squared minus 2 cubed .

```
In[28]:=  (3 ^ 2) - (2 ^ 3)
Out[28]=  1
```

+1.7 Compute 2 cubed times 3 squared

```
In[29]:=  (2 ^ 3) * (3 ^ 2)
Out[29]=  72
```

+1.8 Compute "double the sum of eight and negative eleven"

```
In[30]:= (8 - 11) * 2
Out[30]= -6
```

Exercises for Section 2 | Introducing Functions

2.1 Compute 7 + 6 + 5 using the function Plus

```
In[31]:= Plus[7, 6, 5]
Out[31]= 18
```

2.2 Compute 2 x (3 + 4) using Times and Plus

```
In[32]:= Times[2, Plus[3, 4]]
Out[32]= 14
```

2.3 Use Max to find the larger of 6x8 and 5x9

```
In[33]:= Max[Times[6, 8], Times[5, 9]]
Out[33]= 48
```

2.4 Use RandomInteger to generate a random number between 0 and 1000.

```
In[34]:= RandomInteger[1000]
Out[34]= 443
```

2.5 Use Plus and RandomInteger to generate a number between 10 and 20.

```
In[35]:= Plus[10, RandomInteger[10]]
Out[35]= 12
```

+2.1 Compute 5 x4x3x2 using Times .

```
In[36]:= Times[5, 4, 3, 2]
Out[36]= 120
```

+2.2 Compute 2 - 3 using Subtract

```
In[37]:= Subtract[2, 3]
Out[37]= -1
```

+2.3 Compute (8 + 7) + (9 + 2) using Times and Plus

```
In[38]:=  Times[Plus[8, 7], Plus[9, 2]]
Out[38]=  165
```

+2.4 Compute (26 - 89)/9 using Subtract and Divide

```
In[39]:=  Divide[Subtract[26, 89], 9]
Out[39]=  -7
```

+2.5 Compute 100 - 5^2 using Subtract and Power

```
In[40]:=  Subtract[100, Power[5, 2]]
Out[40]=  75
```

+2.6 Find the larger of 3^5 and 5^3

```
In[41]:=  Max[Power[3, 5], Power[5, 3]]
Out[41]=  243
```

+2.7 Multiply 3 and the larger of 4^3 and 3^4

```
In[42]:=  Times[3, Max[Power[4, 3], Power[3, 4]]]
Out[42]=  243
```

+2.8 Add two random numbers each between 0 and 1000.

```
In[43]:=  Plus[RandomInteger[1000], RandomInteger[1000]]
Out[43]=  698
```

Exercises for Section 3 | First Look at Lists

3.1 Use Range to create the list {1, 2, 3, 4}

```
In[44]:=  Range[4]
Out[44]=  {1, 2, 3, 4}
```

3.2 Make a list of numbers up to 100

```
In[45]:=  Range[100]
Out[45]=  {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23,
           24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43,
           44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62,
           63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81,
           82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100}
```

3.3 Use Range and Reverse to create {4, 3, 2, 1}

```
In[46]:= Reverse[Range[4]]
Out[46]= {4, 3, 2, 1}
```

3.4 Make a list of numbers from 1 to 50 in reverse order .

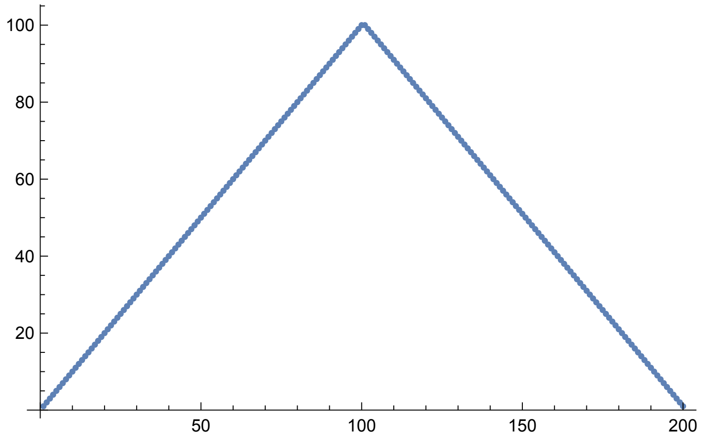
```
In[47]:= Reverse[Range[50]]
Out[47]= {50, 49, 48, 47, 46, 45, 44, 43, 42, 41, 40, 39, 38, 37,
          36, 35, 34, 33, 32, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20,
          19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1}
```

3.5 Use Range, Reverse and Join to create {1, 2, 3, 4, 4, 3, 2, 1}

```
In[48]:= Join[Range[4], Reverse[Range[4]]]
Out[48]= {1, 2, 3, 4, 4, 3, 2, 1}
```

3.6 Plot a list that counts up from 1 to 100, then down to 1.

```
In[49]:= ListPlot[Join[Range[100], Reverse[Range[100]]]]
Out[49]=
```



3.7 Use Range and RandomInteger to make a list with a random lenght up to 10.

```
In[51]:= Range[RandomInteger[10]]
Out[51]= {1, 2, 3}
```

3.8 Find a simpler form for Reverse[Reverse[Range[10]]]

```
In[52]:= Range[10]
Out[52]= {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

3.9 Find a simpler form to Join[{1, 2}, Join[{3, 4}, {5}]]

```
In[53]:=  Range[5]
Out[53]=  {1, 2, 3, 4, 5}
```

3.10 Find a simpler form for Join[Range[10], Join[Range[10], Range[5]]]

```
In[54]:=  Join[Range[10], Range[10], Range[5]]
Out[54]=  {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1, 2, 3, 4, 5}
```

3.11 Find a simpler form for Reverse[Join[Range[20], Reverse[Range[20]]]]

```
In[55]:=  Join[Range[20], Reverse[Range[20]]]
Out[55]=  {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
           20, 20, 19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1}
```

+3.1 Compute the reverse of the reverse of {1, 2, 3, 4}

```
In[57]:=  Reverse[Reverse[Range[4]]]
Out[57]=  {1, 2, 3, 4}
```

+3.2 Use Range, Reverse and Join to create the list {1, 2, 3, 4, 5, 4, 3, 2, 1} .

```
In[58]:=  Join[Range[4], Reverse[Range[4]]]
Out[58]=  {1, 2, 3, 4, 4, 3, 2, 1}
```

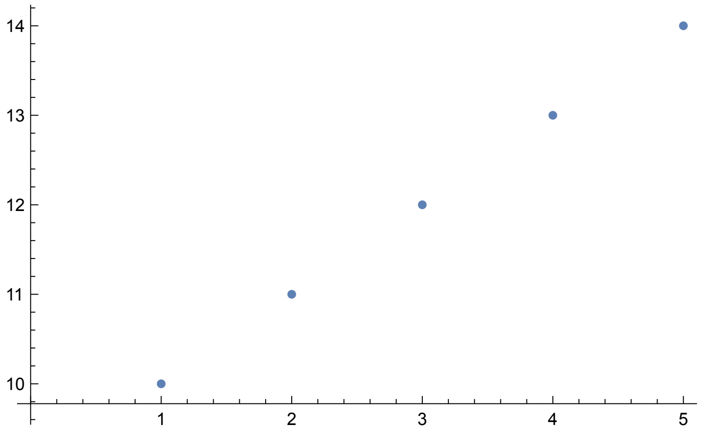
+3.3 Use Range, Reverse and Join to create {3, 2, 1, 4, 3, 2, 1, 5, 4, 3, 2, 1}

```
In[59]:=  Join[Reverse[Range[3]], Reverse[Range[4]], Reverse[Range[5]]]
Out[59]=  {3, 2, 1, 4, 3, 2, 1, 5, 4, 3, 2, 1}
```

+3.4 Plot the list numbers {10, 11, 12, 13, 14}

```
In[60]:=  ListPlot[Range[10, 14]]
```

Out[60]=

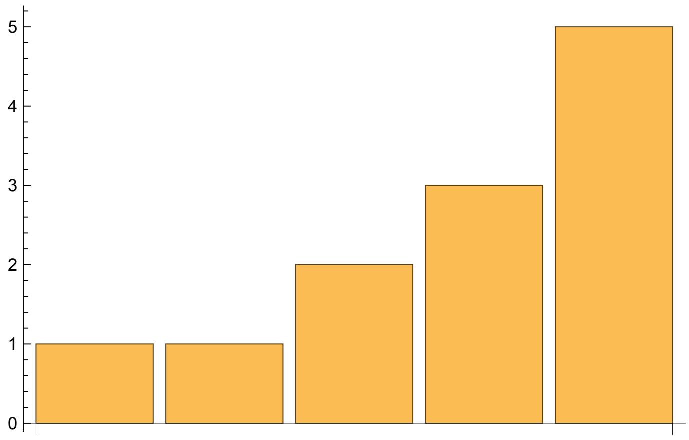


Exercises for Section 4 | Displaying Lists

4.1 Make a bar chart of {1, 1, 2, 3, 5}

In[61]:=
```
BarChart[{1, 1, 2, 3, 5}]
```

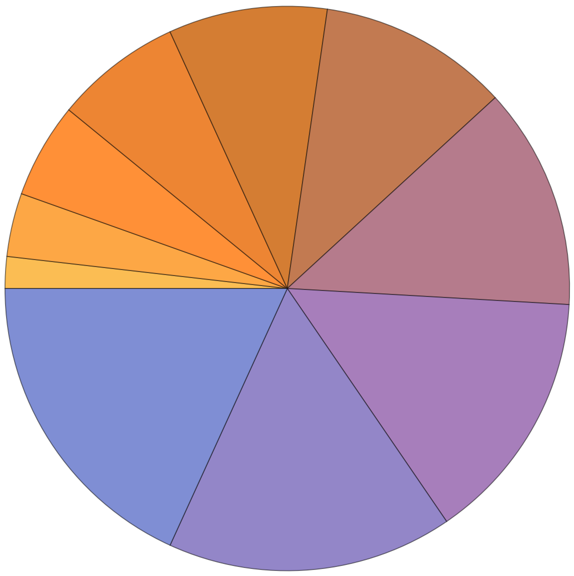
Out[61]=



4.2 Make a pie chart of numbers from 1 to 10.

In[62]:=
```
PieChart[Range[10]]
```

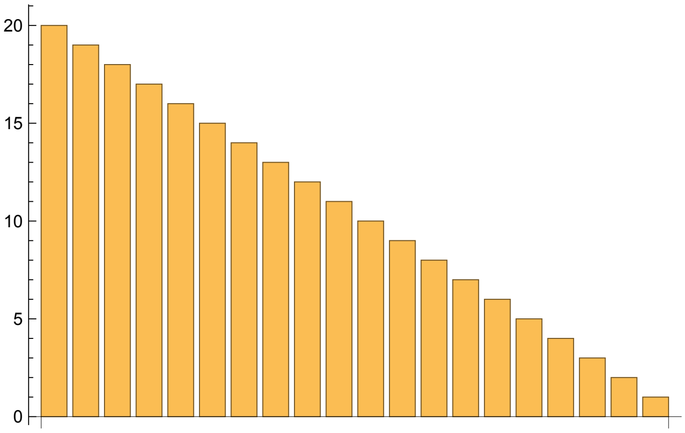
Out[62]=



4.3 Make a bar chart of numbers counting down from 20 to 1.

```
In[63]:= BarChart[Reverse[Range[20]]]
```

Out[63]=



4.4 Display numbers from 1 to 5 in a column .

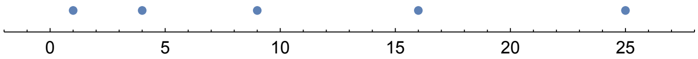
```
In[64]:= Column[Range[5]]
```

```
         1
         2
Out[64]= 3
         4
         5
```

4.5 Make a number line plot of the squares {1, 4, 9, 16, 25} .

```
In[65]:= NumberLinePlot[Range[5]^2]
```

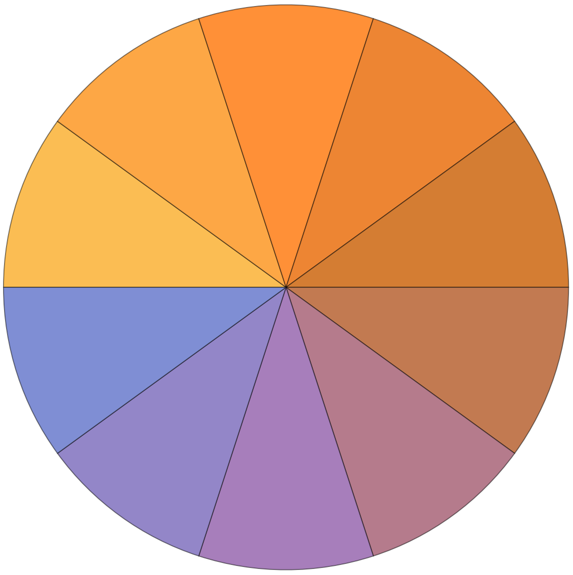
Out[65]=



4.6 Make a pie chart with 10 identical segments, each of size 1.

```
In[66]:= PieChart[ConstantArray[1, 10]]
```

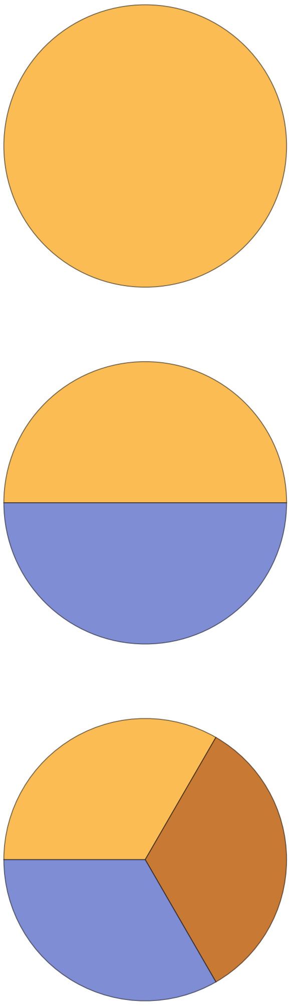
Out[66]=



4.7 Make a column of pie charts with 1, 2 and 3 identical segments.

```
In[67]:= Column[{PieChart[{1}], PieChart[{1, 1}], PieChart[{1, 1, 1}]}, 3]
```

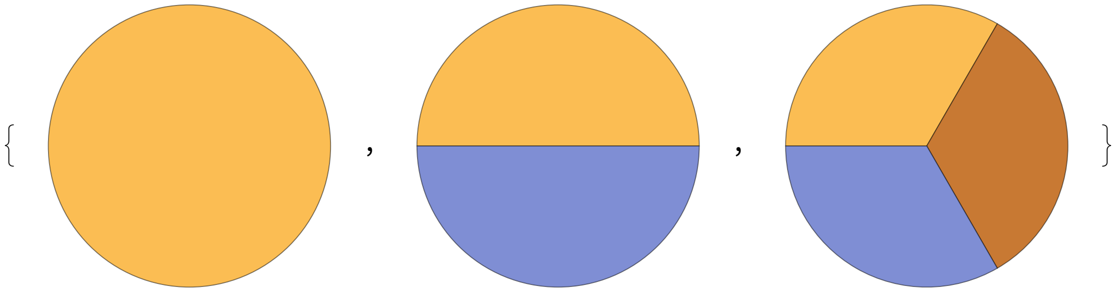
Out[67]=



+4.1 Make a list of pie charts with 1, 2 and 3 identical segments .

```
In[68]:= {PieChart[{1}], PieChart[{1, 1}], PieChart[{1, 1, 1}]}
```

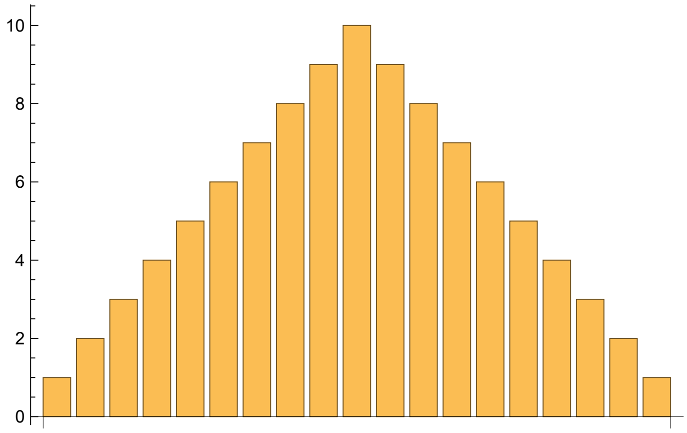
Out[68]=



+4.2 Make a bar chart of the sequence 1, 2, 3, ..., 9, 10, 9, 8, 7, ..., 1.

```
In[69]:= BarChart[Join[Range[10], Reverse[Range[9]]]]
```

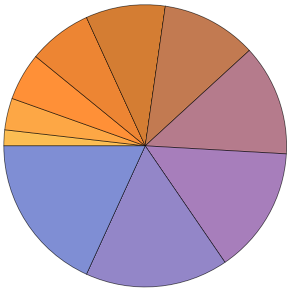
Out[69]=



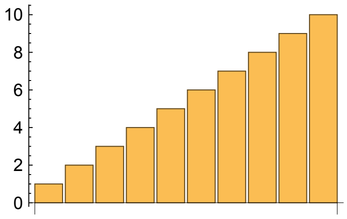
+4.3 Make a list of pie chart, bar chart and line plot of the numbers from 1 to 10.

```
In[70]:= {PieChart[Range[10]], BarChart[Range[10]], ListLinePlot[Range[10]]}
```

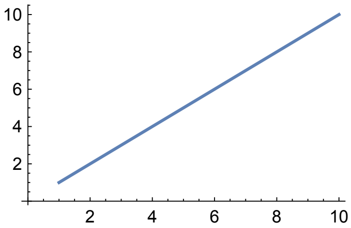
Out[70]= {



,



,

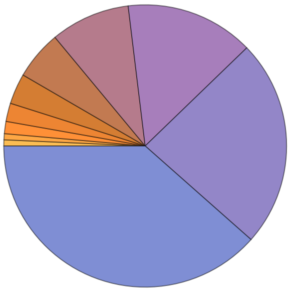


}

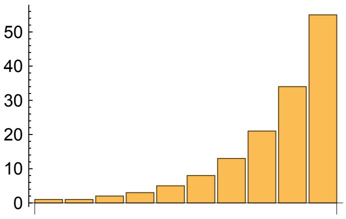
+4.4 Make a list of a pie chart and a bar chart of {1, 1, 2, 3, 5, 8, 13, 21, 34, 55}

```
In[71]:= {PieChart[Fibonacci[Range[10]]], BarChart[Fibonacci[Range[10]]]}
```

Out[71]= {



,

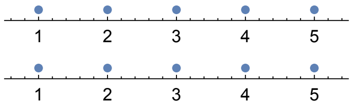


}

+4.5 Make a column of two number line plot of {1, 2, 3, 4, 5}

```
In[72]:= Column[{NumberLinePlot[Range[5]], NumberLinePlot[Range[5]]}]
```

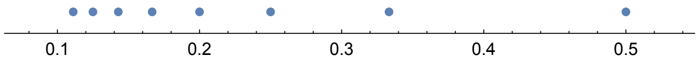
Out[72]=



+4.6 Mae a number line of fractions 1/2, 1/3, ... through 1/9.

```
In[73]:= NumberLinePlot[{ConstantArray[1, 8] / Range[2, 9]}]
```

Out[73]=



Exercises for Section 5 | Operations on Lists

5.1 Make a list of the first 10 squares, in reverse order .

```
In[74]:= Reverse[Range[10]^2]
Out[74]= {100, 81, 64, 49, 36, 25, 16, 9, 4, 1}
```

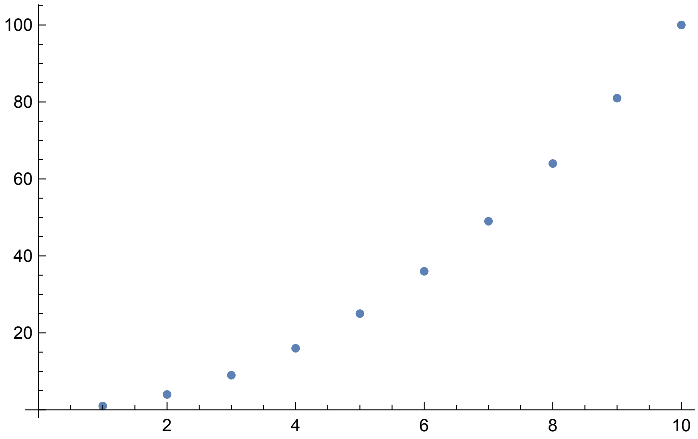
5.2 Find the total of the first 10 squares .

```
In[75]:= Total[Reverse[Range[10]^2]]
Out[75]= 385
```

5.3 Make a plot of the first 10 squares, starting at 1.

```
In[76]:= ListPlot[Range[10]^2]
```

Out[76]=



5.4 Use Sort, Join and Range to create {1, 1, 2, 2, 3, 3, 4, 4}.

```
In[78]:= Sort[Join[Range[4], Range[4]]]
Out[78]= {1, 1, 2, 2, 3, 3, 4, 4}
```

5.5 Use Range and + to make a list of numbers from 10 to 20, inclusive .

```
In[79]:=  Range[0, 10] + 10
Out[79]=  {10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20}
```

5.6 Make a combined list of the first 5 squares and cubes (numbers raised to the power 3), sorted into order .

```
In[80]:=  Sort[Join[Range[5]^2, Range[5]^3]]
Out[80]=  {1, 1, 4, 8, 9, 16, 25, 27, 64, 125}
```

5.7 Find the number of digits in 2^128.

```
In[81]:=  Length[IntegerDigits[2^128]]
Out[81]=  39
```

5.8 Find the first digit of 3^32

```
In[82]:=  First[IntegerDigits[2^32]]
Out[82]=  4
```

5.9 Find the first 10 digits in 2^100.

```
In[83]:=  Take[IntegerDigits[2^100], 10]
Out[83]=  {1, 2, 6, 7, 6, 5, 0, 6, 0, 0}
```

5.10 Find the largest digit that appears in 2^20.

```
In[84]:=  Max[IntegerDigits[2^20]]
Out[84]=  8
```

5.11 Find how many zeros appear in the digits of 2^1000.

```
In[85]:=  Count[IntegerDigits[2^1000], 0]
Out[85]=  28
```

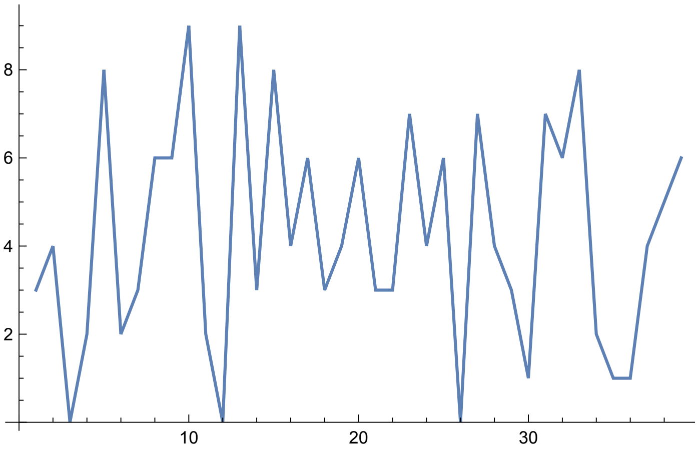
5.12 Use Part, Sort and IntegerDigit to find the second - smallest digit in 2^20.

```
In[86]:=  Part[Sort[IntegerDigits[2^20]], 2]
Out[86]=  1
```

5.13 Make a line plot of the sequence of digits that appear in 2^128.

```
In[87]:=  ListLinePlot[IntegerDigits[2^128]]
```

Out[87]=



5.14 Use Take and Drop to get the sequence 11 through 20 from Range[100].

```
In[88]:=  Take[Drop[Range[100], 10], 10]
Out[88]=  {11, 12, 13, 14, 15, 16, 17, 18, 19, 20}
```

+5.1 Make a list of the first 10 multiples of 3.

```
In[89]:=  Range[10] * 3
Out[89]=  {3, 6, 9, 12, 15, 18, 21, 24, 27, 30}
```

+5.2 Make a list of the first 10 squares using only Range and Times .

```
In[90]:=  Times[Range[10], Range[10]]
Out[90]=  {1, 4, 9, 16, 25, 36, 49, 64, 81, 100}
```

+5.3 Find the last digit of 2^37.

```
In[91]:=  Last[IntegerDigits[2^37]]
Out[91]=  2
```

+5.4 Find the second-to-last digit of 2^32.

```
In[92]:=  Last[Drop[IntegerDigits[2^32], -1]]
Out[92]=  9
```

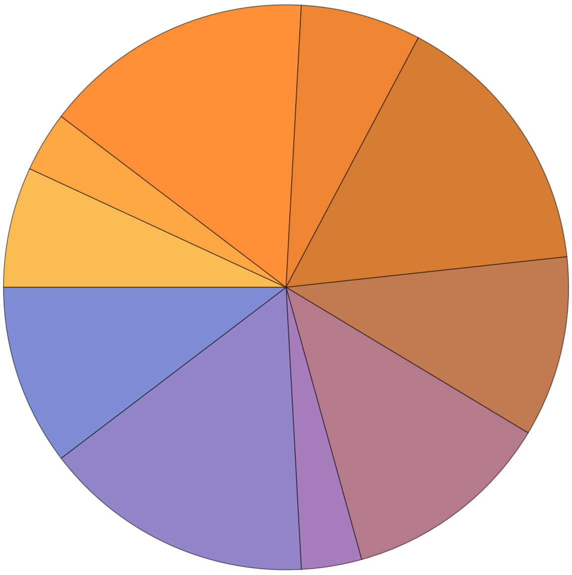
+5.5 Find the sum of all the digits of 3^126.

```
In[93]:=  Total[IntegerDigits[3^126]]
Out[93]=  234
```

+5.6 Make a pie chart of the sequence of digits that appear in 2^32.

```
In[94]:= PieChart[IntegerDigits[2^32]]
```

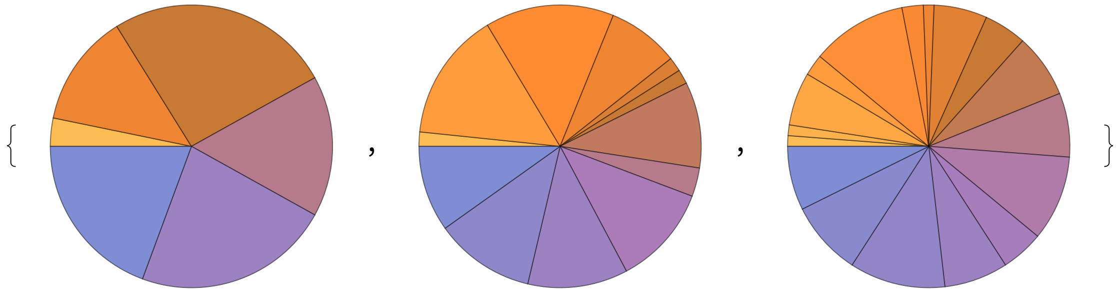
Out[94]=



+5.7 Make a list of pie charts for the sequence of digits in 2^20, 2^40, 2^60.

```
In[95]:= {PieChart[IntegerDigits[2^20]],
          PieChart[IntegerDigits[2^40]], PieChart[IntegerDigits[2^60]]}
```

Out[95]=



Exercises for Section 6 | Making Tables

6.1 Make a list in which the number 1000 is repeated 5 times .

```
In[96]:= Table[1000, {5}]
Out[96]= {1000, 1000, 1000, 1000, 1000}
```

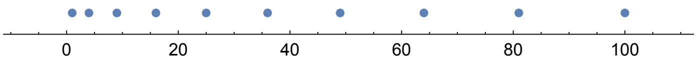
6.2 Make a table of the value of n^3 for n from 10 to 20.

```
In[97]:=  Table[n^3, {n, 10, 20}]
Out[97]=  {1000, 1331, 1728, 2197, 2744, 3375, 4096, 4913, 5832, 6859, 8000}
```

6.3 Make a number line plot of the first 20 squares .

```
In[99]:=  NumberLinePlot[Table[n^2, {n, 10}]]
```

Out[99]=



6.4 Make a list of the even numbers (2, 4, 6, ...) up to 20.

```
In[98]:=  Table[n*2, {n, 10}]
Out[98]=  {2, 4, 6, 8, 10, 12, 14, 16, 18, 20}
```

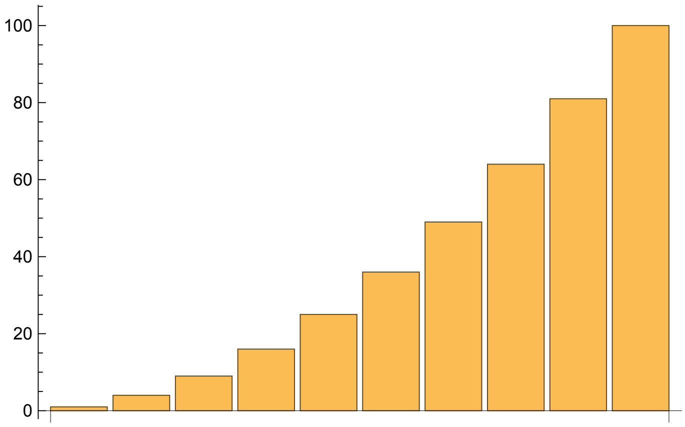
6.5 Use Table to get the same result as Range[10].

```
In[100]:=  Table[n, {n, 10}]
Out[100]=  {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

6.6 Make a bar chart of the first 10 squares .

```
In[101]:=  BarChart[Table[n^2, {n, 10}]]
```

Out[101]=



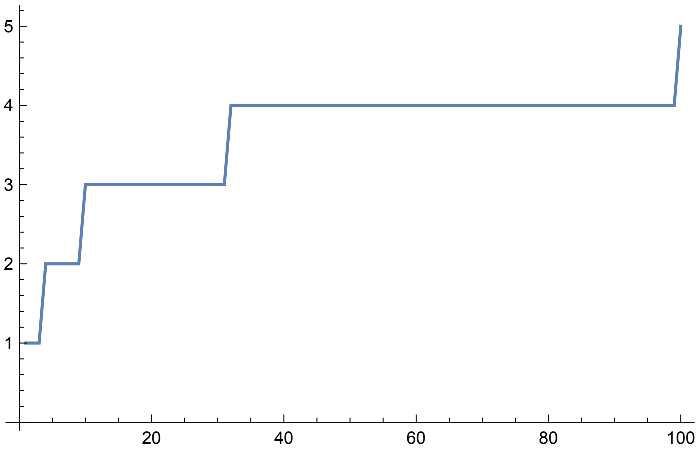
6.7 Make a table of lists of digits for the first 10 squares.

```
In[102]:=  Table[IntegerDigits[n^2], {n, 10}]
Out[102]=  {{1}, {4}, {9}, {1, 6}, {2, 5}, {3, 6}, {4, 9}, {6, 4}, {8, 1}, {1, 0, 0}}
```

6.8 Make a list line plot of the length of the sequence of digits for each of the first 100 squares.

```
In[103]:= ListLinePlot[Table[Length[IntegerDigits[n^2]], {n, 100}]]
```

Out[103]=



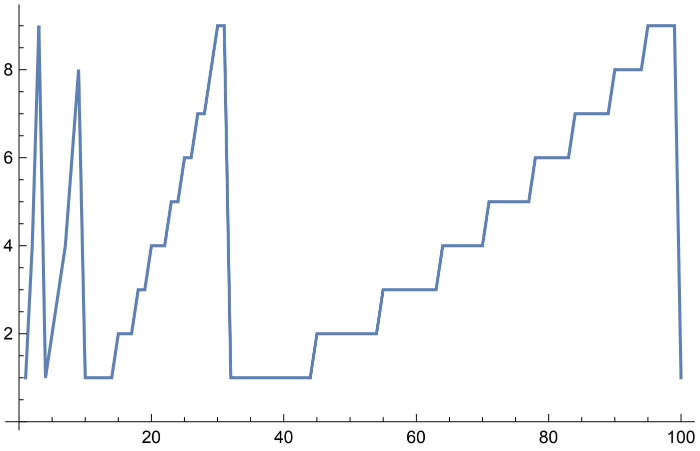
6.9 Make a table of the first digit of the first 20 squares.

```
In[104]:= Table[First[IntegerDigits[n^2]], {n, 20}]
Out[104]= {1, 4, 9, 1, 2, 3, 4, 6, 8, 1, 1, 1, 1, 1, 2, 2, 2, 3, 3, 4}
```

6.10 Make a list line plot of the first digits of the first 100 squares.

```
In[105]:= ListLinePlot[Table[First[IntegerDigits[n^2]], {n, 100}]]
```

Out[105]=



+6.1 Make a list of the differences between n^3 and n^2 with n up to 10.

```
In[106]:= Table[(n^3) - (n^2), {n, 10}]
Out[106]= {0, 4, 18, 48, 100, 180, 294, 448, 648, 900}
```

+6.2 Make a list of the odd numbers (1, 3, 5, ...) up to 100.

```
In[107]:= Table[(n * 2) - 1, {n, 50}]

Out[107]= {1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31,
           33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65,
           67, 69, 71, 73, 75, 77, 79, 81, 83, 85, 87, 89, 91, 93, 95, 97, 99}
```

+6.3 Make a list of the squares of even numbers up to 100.

```
In[108]:= Table[(n * 2) ^2, {n, 50}]

Out[108]= {4, 16, 36, 64, 100, 144, 196, 256, 324, 400, 484, 576, 676, 784, 900,
           1024, 1156, 1296, 1444, 1600, 1764, 1936, 2116, 2304, 2500, 2704, 2916,
           3136, 3364, 3600, 3844, 4096, 4356, 4624, 4900, 5184, 5476, 5776, 6084,
           6400, 6724, 7056, 7396, 7744, 8100, 8464, 8836, 9216, 9604, 10 000}
```

+6.4 Create the list {-3, -2, -1, 0, 1, 2} using Range .

```
In[109]:= Range[-3, 2]

Out[109]= {-3, -2, -1, 0, 1, 2}
```

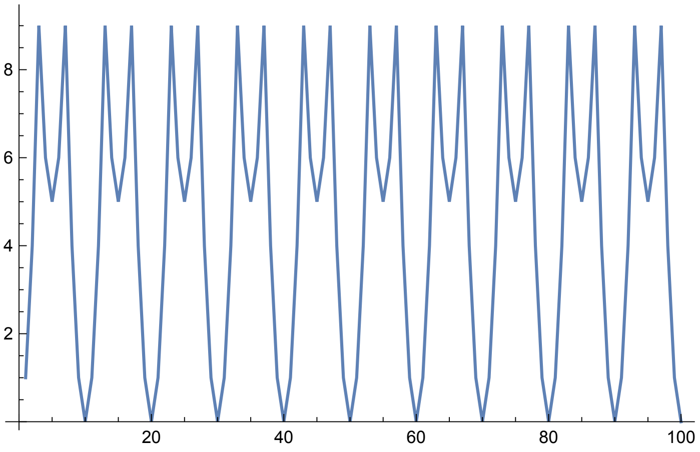
+6.5 Make a list for the numbers n up to 20 in which each element is a column of the value of n, n^2 and n^3.

```
In[110]:= Table[Column[{n, n^2, n^3}], {n, 20}]

            1   2   3    4    5     6     7     8     9     10     11
Out[110]= { 1 , 4 , 9  , 16 , 25  , 36  , 49  , 64  , 81  , 100  , 121  ,
            1   8   27   64   125   216   343   512   729   1000   1331

           12     13     14     15     16     17     18     19     20
           144  , 169  , 196  , 225  , 256  , 289  , 324  , 361  , 400  }
           1728   2197   2744   3375   4096   4913   5832   6859   8000
```

+6.6 Make a list line plot of the last digit of the first 100 squares.

```
In[111]:= ListLinePlot[Table[Last[IntegerDigits[n^2]], {n, 100}]]
```

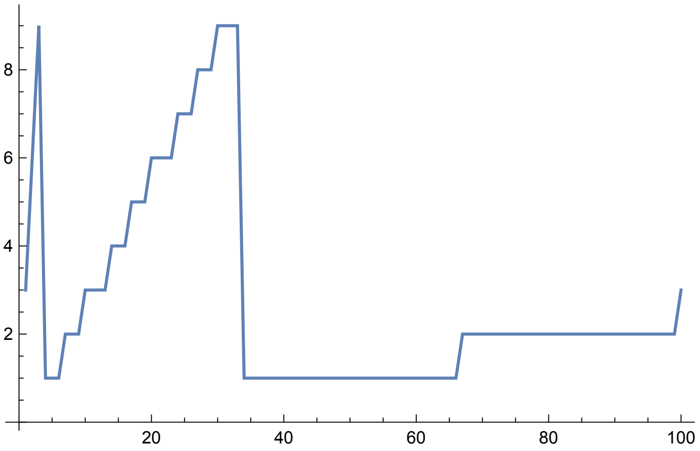
Out[111]=



+6.7 Make a list line plot of the first digit of the first 100 multiples of 3.

```
In[112]:= ListLinePlot[Table[First[IntegerDigits[n * 3]], {n, 100}]]
```

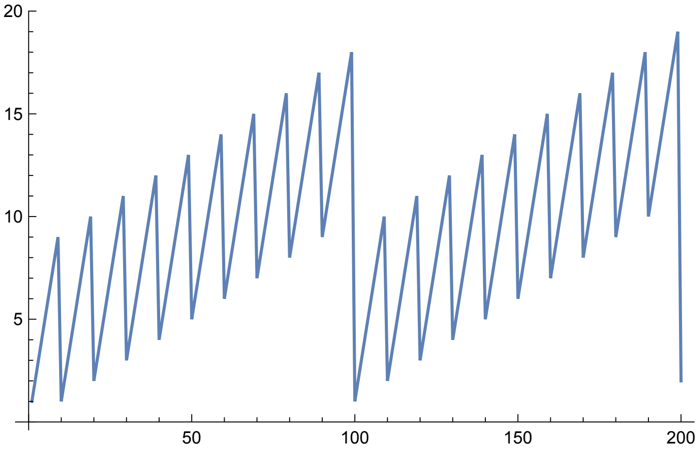
Out[112]=



+6.8 Make a list line plot of the total of the digits for each number up to 200.

```
In[113]:= ListLinePlot[Table[Total[IntegerDigits[n]], {n, 200}]]
```

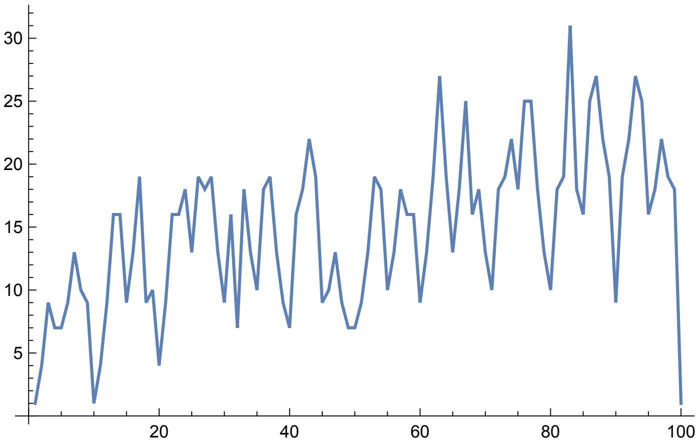
Out[113]=



+6.9 Make a list line plot of the total of the digits for each of the first 100 squares.

```
In[114]:= ListLinePlot[Table[Total[IntegerDigits[n^2]], {n, 100}]]
```

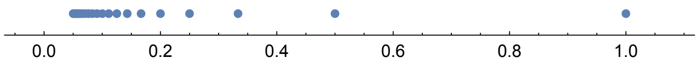
Out[114]=



+6.10 Make a number line plot of the numbers 1/n with n from 1 to 20.

```
In[115]:= NumberLinePlot[Table[1 / n, {n, 1, 20}]]
```

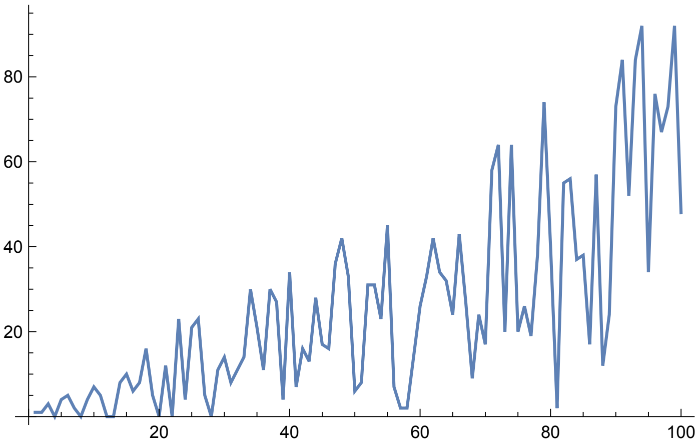
Out[115]=



+6.11 Make a line plot of a list of 100 random integers where the nth integer is between 0 and n.

```
In[116]:= ListLinePlot[Table[RandomInteger[n], {n, 100}]]
```

Out[116]=



Exercises for Section 7 | Colors and Styles

7.1 Make a list of red, yellow and green.

```
In[117]:= {Red, Yellow, Green}
```

Out[117]= {🟥, 🟨, 🟩}

7.2 Make a red, yellow, green column ("traffic light")

```
In[118]:= Column[{Red, Yellow, Green}]

           🟥
Out[118]=  🟨
           🟩
```

7.3 Compute the negation of the color Orange.

```
In[119]:= ColorNegate[Orange]

Out[119]= 🟦
```

7.4 Make a list of color with hues varying from 0 to 1 in steps of 0.02.

```
In[120]:= Table[Hue[n], {n, 0, 1, 0.02}]

Out[120]= {🟥, 🟥, 🟥, 🟧, 🟧, 🟧, 🟨, 🟨, 🟨, 🟨, 🟨, 🟩, 🟩, 🟩, 🟩,
           🟩, 🟩, 🟩, 🟩, 🟩, 🟩, 🟩, 🟩, 🟩, 🟦, 🟦, 🟦, 🟦, 🟦, 🟦, 🟦, 🟦, 🟦,
           🟦, 🟦, 🟦, 🟪, 🟪, 🟪, 🟪, 🟪, 🟪, 🟪, 🟪, 🟥, 🟥, 🟥, 🟥, 🟥, 🟥, 🟥}
```

7.8 Make a list of numbers from 0 to 1 in steps of 0.1, each with a hue equal to its value.

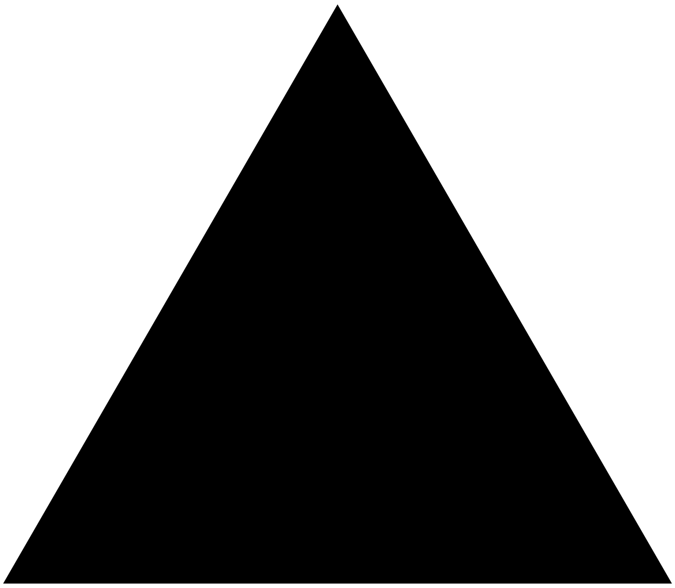
```
In[122]:= Table[{Hue[n], Style[n, Hue[n]]}, {n, 0, 1, 0.1}]

Out[122]= {{🟥, 0.}, {🟧, 0.1}, {🟨, 0.2}, {🟩, 0.3}, {🟩, 0.4},
           {🟦, 0.5}, {🟦, 0.6}, {🟦, 0.7}, {🟪, 0.8}, {🟥, 0.9}, {🟥, 1.}}
```

Exercises for Section 8 | Basic Graphics Objects

8.1 Use RegularPolygon to draw a triangle.

```
In[123]:= Graphics[RegularPolygon[3]]
```

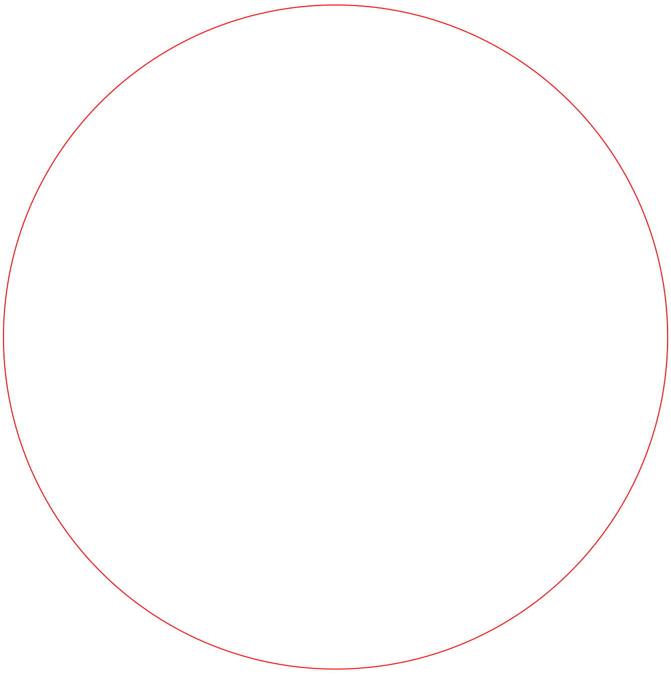
Out[123]=



8.2 Make graphics of a red circle.

```
In[124]:= Graphics[Style[Circle[], Red]]
```

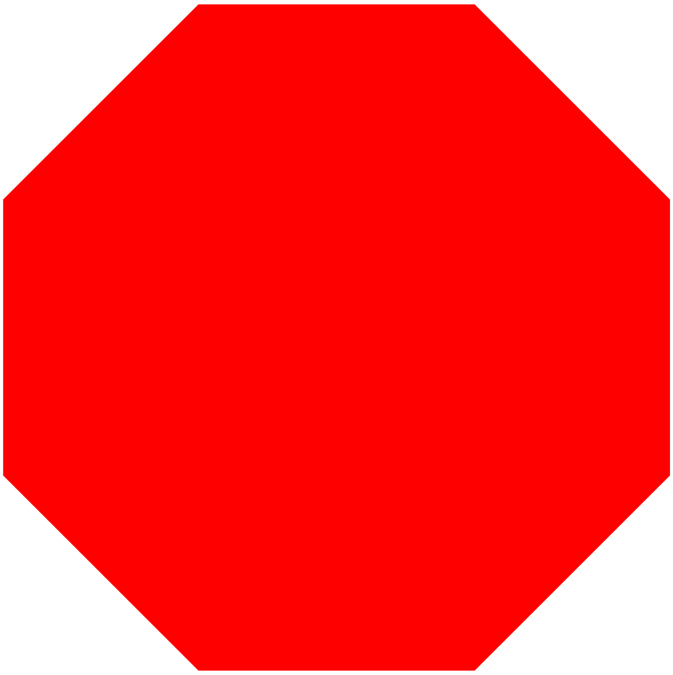
Out[124]=



8.3 Make a red octagon.

```
In[125]:= Graphics[Style[RegularPolygon[8], Red]]
```

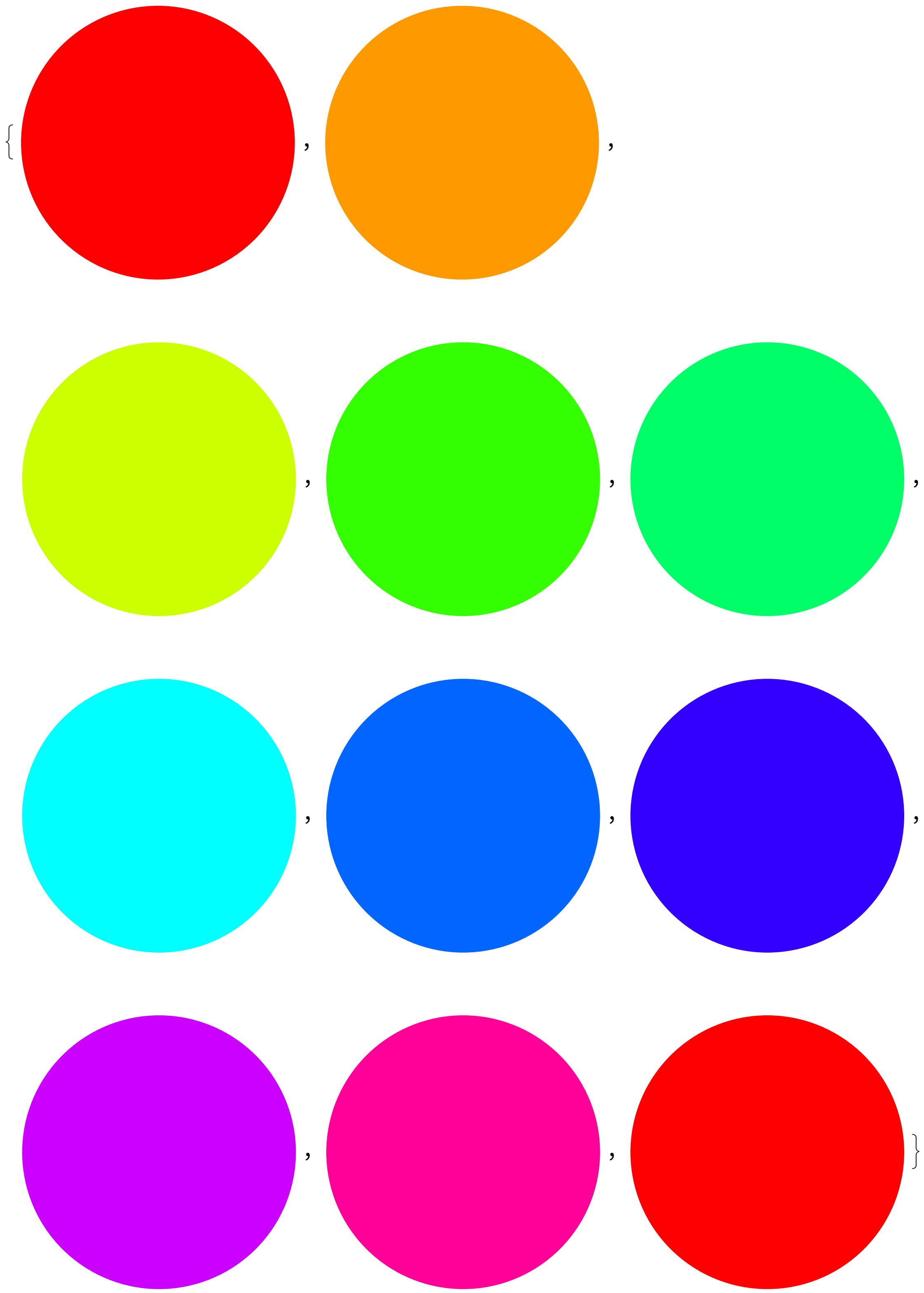
Out[125]=



8.4 Make a list whose elements are disks with hues varying from 0 to 1 in steps of 0.1.

```
In[127]:= Table[Graphics[Style[Disk[], Hue[n]]], {n, 0, 1, 0.1}]
```

Out[127]=

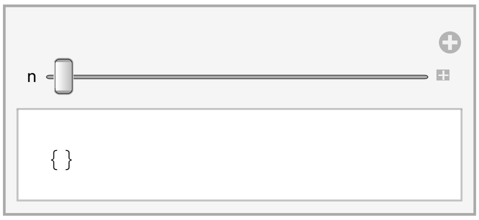


Exercise for Section 9 | Interactive Manipulation

9.1 Make a Manipulate to show Range[n] with n varying from 0 to 100.

In[128]:=
```
Manipulate[Range[n], {n, 0, 100}]
```

Out[128]=

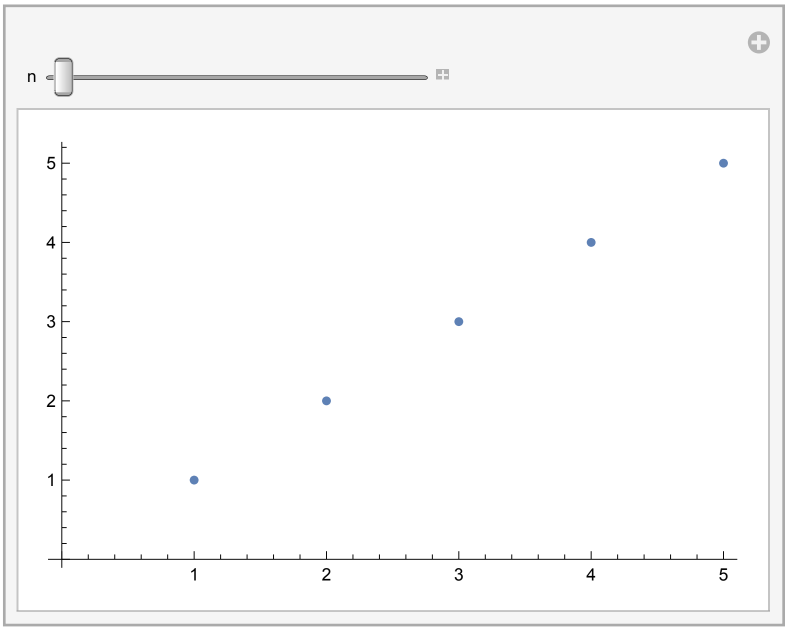


```
{ }
```

9.2 Make a Manipulate to plot the whole numbers up to n, where n can range from 5 to 50.

In[129]:=
```
Manipulate[ListPlot[Range[1, n]], {n, 5, 50}]
```

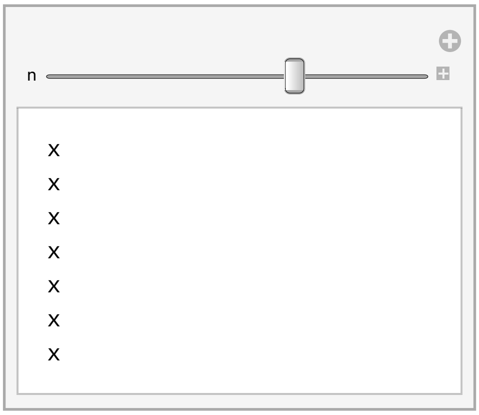
Out[129]=



9.3 Make a Manipulate to show a column of between 1 and 10 copies of x.

```
In[130]:= Manipulate[Column[Table[x, n]], {n, 1, 10, 1}]
```

Out[130]=

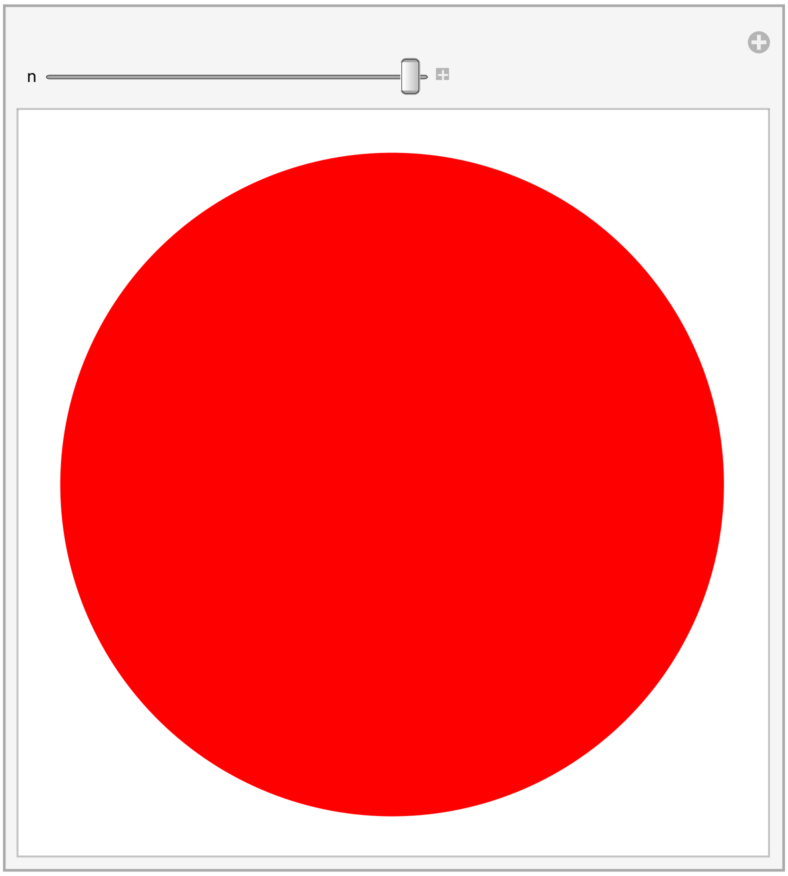


```
x
x
x
x
x
x
x
```

9.4 Make a Manipulate to show a disk with a hue varying from 0 to 1.

```
In[131]:= Manipulate[Graphics[Style[Disk[], Hue[n]]], {n, 0, 1}]
```

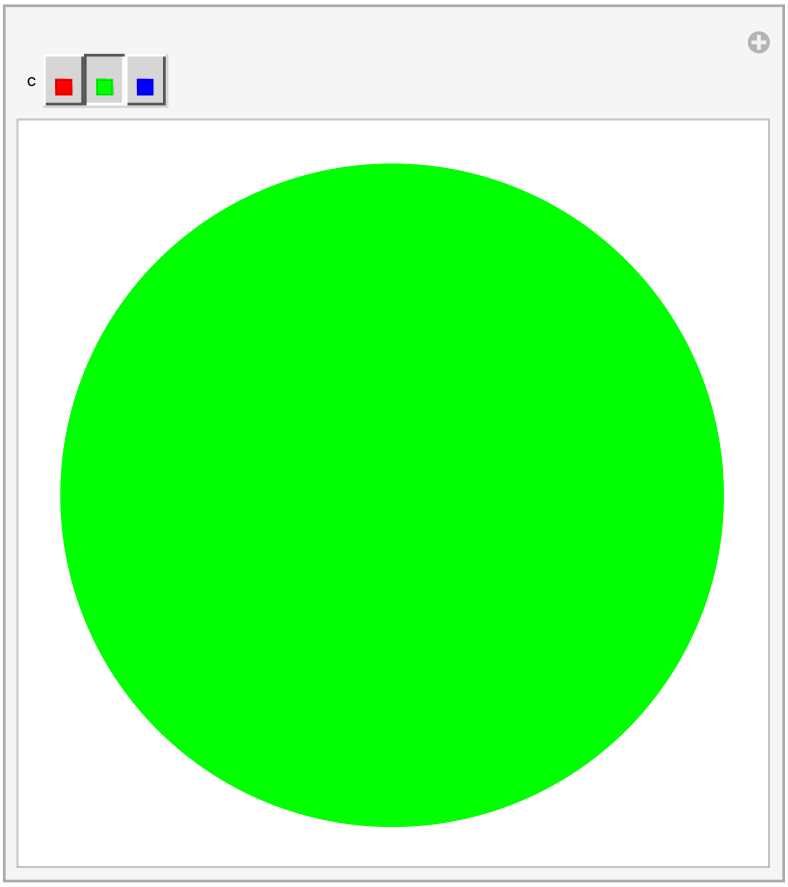
Out[131]=



9.5 Make a manipulate to show a disk with red, green and blue color components varying from 0 to 1.

```
In[132]:= Manipulate[Graphics[Style[Disk[], c]], {c, {Red, Green, Blue}}]
```

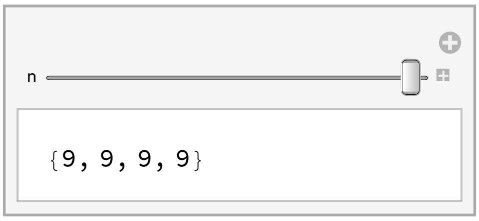
Out[132]=



9.6 Make a Manipulate to show digit sequences of 4 - digit integers (between 1000 and 9999).

```
In[133]:= Manipulate[IntegerDigits[n], {n, 1000, 9999, 1}]
```

Out[133]=

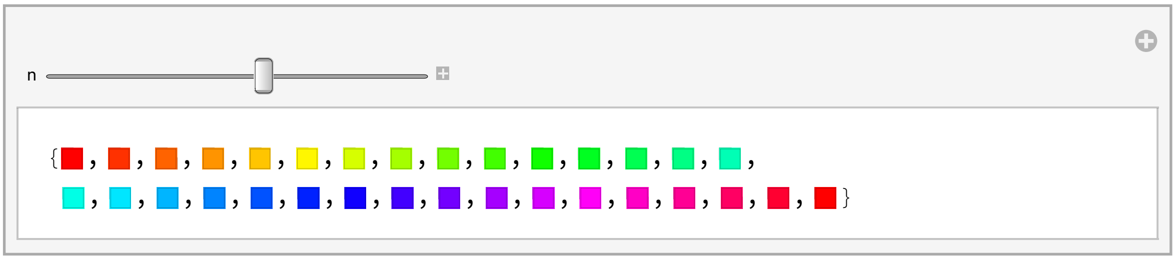


```
{9, 9, 9, 9}
```

9.7 Make a Manipulate to create a list of between 5 and 50 equally spaced hues.

```
In[135]:= Manipulate[Table[Hue[h], {h, 0, 1, 1 / n}], {n, 5, 50, 1}]
```

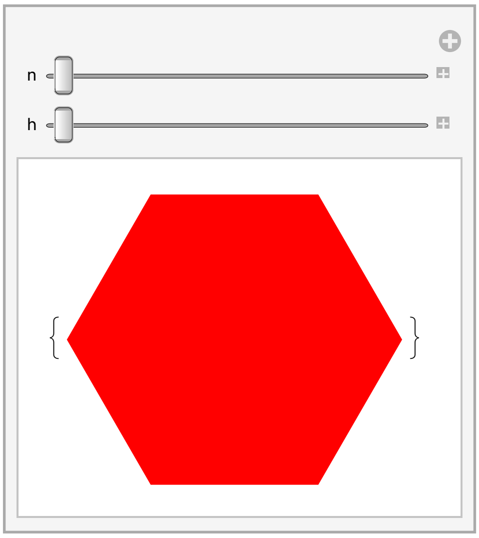
Out[135]=



9.8 Make a Manipulate that shows a list of a variable number of hexagons (between 1 and 10), and with variable hues.

```
In[137]:= Manipulate[Table[Graphics[Style[RegularPolygon[6], Hue[h]]], n],
           {n, 1, 10, 1}, {h, 0, 1}]
```

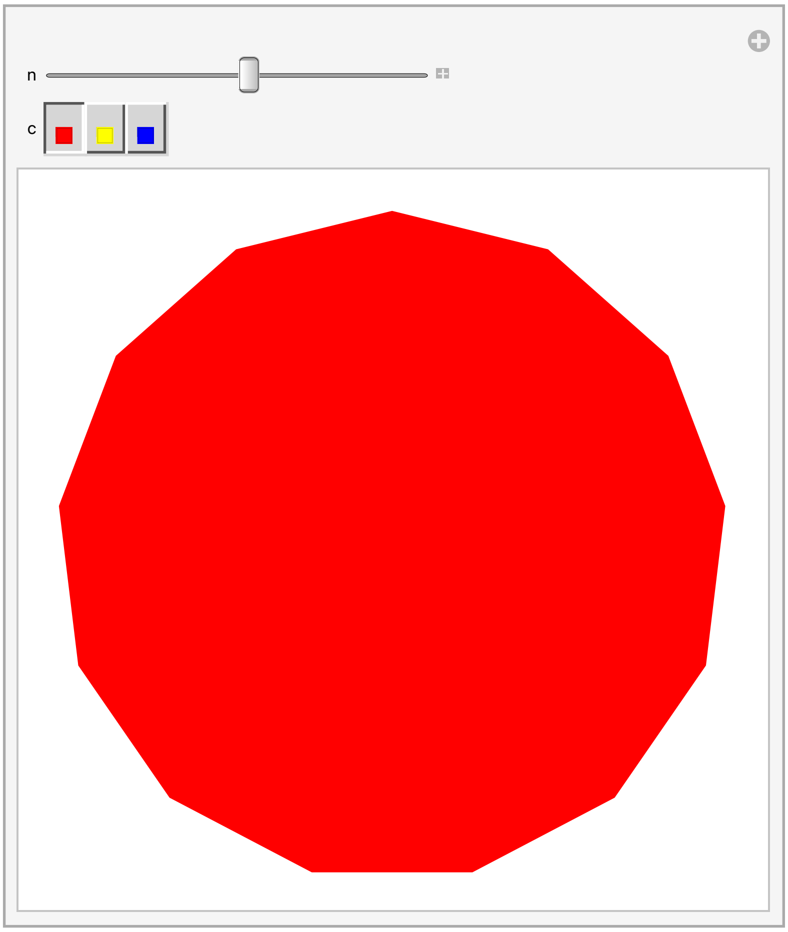
Out[137]=



9.9 Make a Manipulate that lets you show a regular polygon with between 5 and 20 sides, in red, yellow or blue.

In[139]:=
```
Manipulate[Graphics[Style[RegularPolygon[n], c]],
 {n, 5, 20, 1}, {c, {Red, Yellow, Blue}}]
```

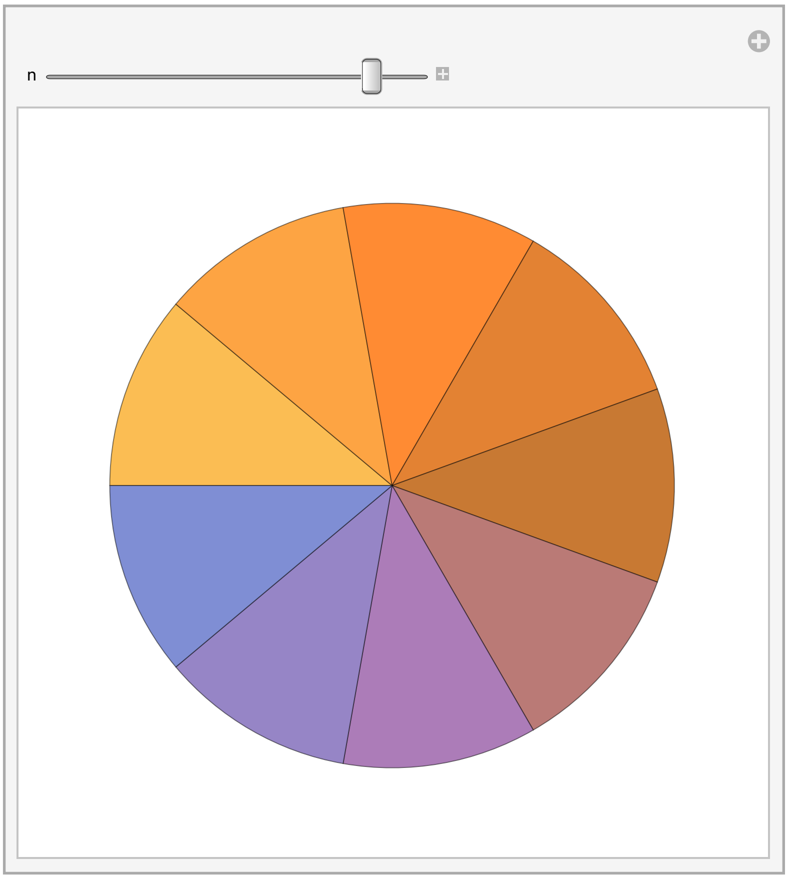
Out[139]=



9.10 Make a Manipulate that shows a pie chart with a number of equal segments varying from 1 to 10.

In[140]:=
```
Manipulate[PieChart[ConstantArray[1, n]], {n, 1, 10, 1}]
```

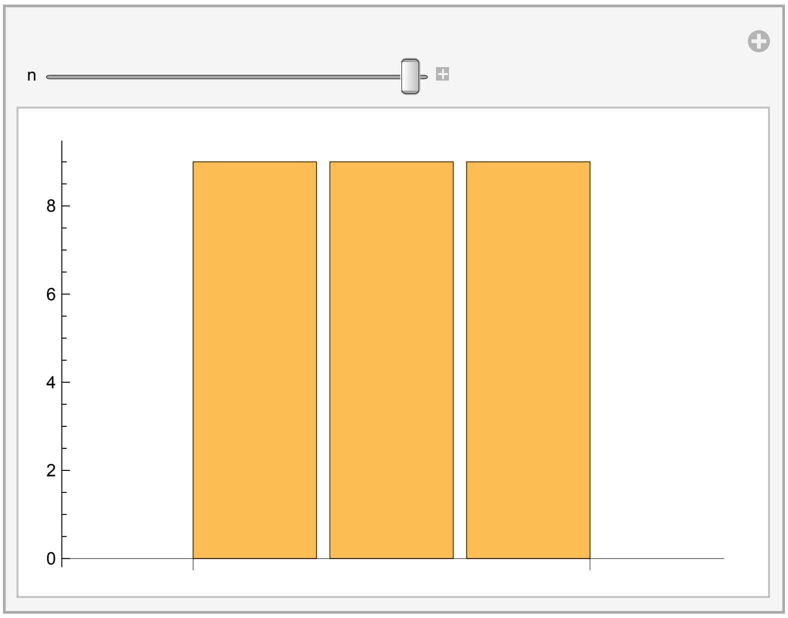
Out[140]=



9.11 Make a Manipulate that gives a bar chart of the 3 digits in integers from 100 to 999.

```
In[141]:= Manipulate[BarChart[IntegerDigits[n]], {n, 100, 999, 1}]
```

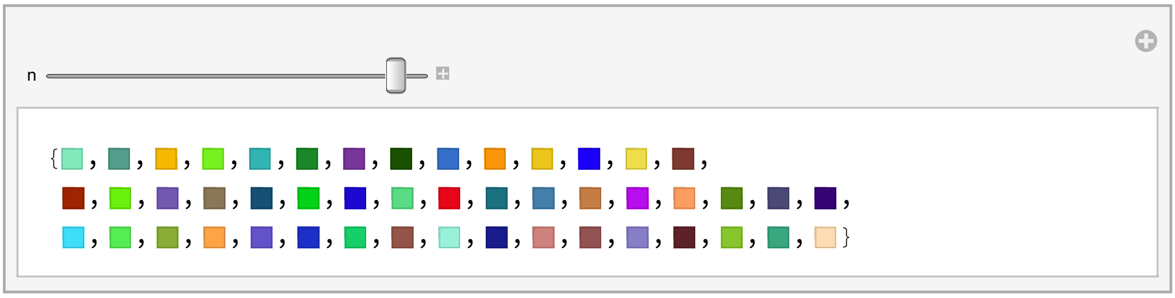
Out[141]=



9.12 Make a Manipulate that shows n random colors, where n can range from 1 to 50.

```
In[142]:= Manipulate[RandomColor[n], {n, 1, 50, 1}]
```

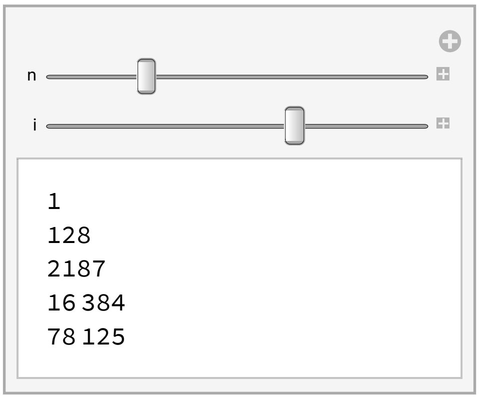
Out[142]=



9.13 Make a Manipulate to display a column of integer powers with bases from 1 to 25 and exponents from 1 to 10.

```
In[143]:= Manipulate[Column[Range[n]^i], {n, 1, 20}, {i, 1, 10, 1}]
```

Out[143]=

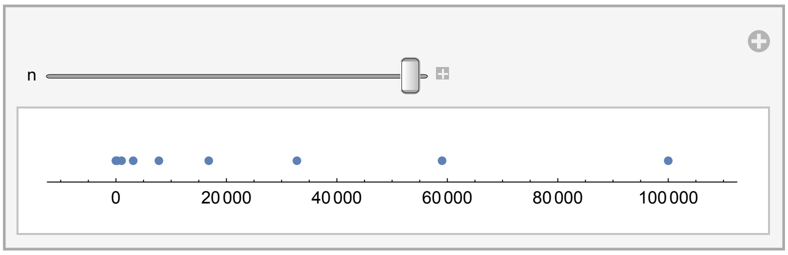


```
1
128
2187
16 384
78 125
```

9.14 Make a Manipulate of a number line of values of x^n for integer x from 1 to 10, with n varying from 0 to 5.

```
In[144]:= Manipulate[NumberLinePlot[Range[10]^n], {n, 0, 5}]
```

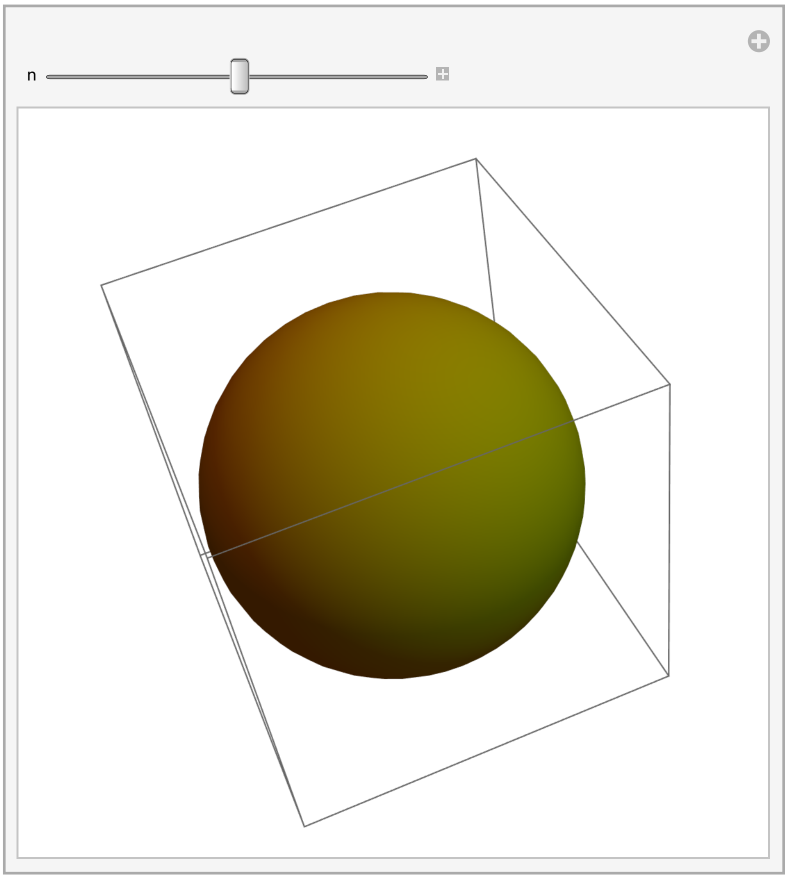
Out[144]=



9.15 Make a Manipulate to show a sphere that can vary in color from green to red.

```
In[145]:= Manipulate[Graphics3D[Style[Sphere[], RGBColor[n, 1 - n, 0]]], {n, 0, 1}]
```

Out[145]=

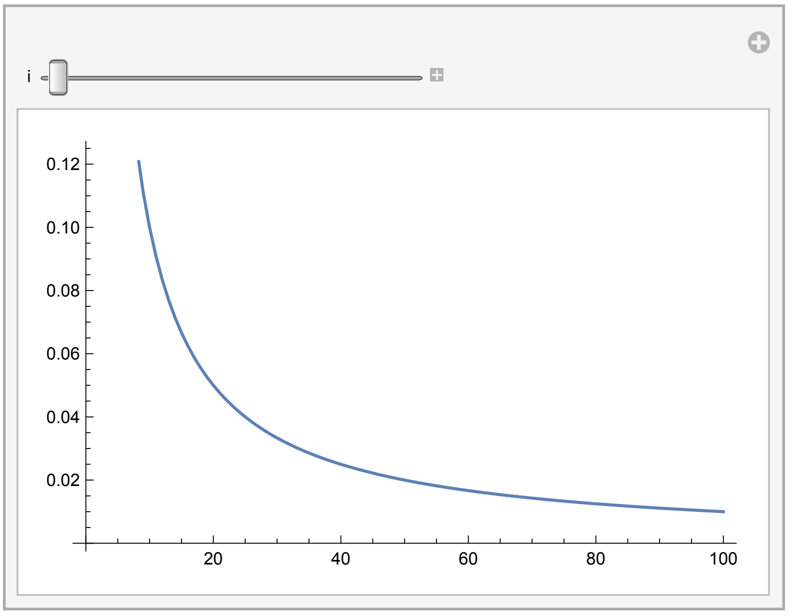


+9.1 Make a Manipulate to plot numbers from 1 to 100 raised to powers that can vary between - 1 and + 1.

In[146]:=

```
Manipulate[ListLinePlot[Range[100]^i], {i, -1, 1}]
```

Out[146]=

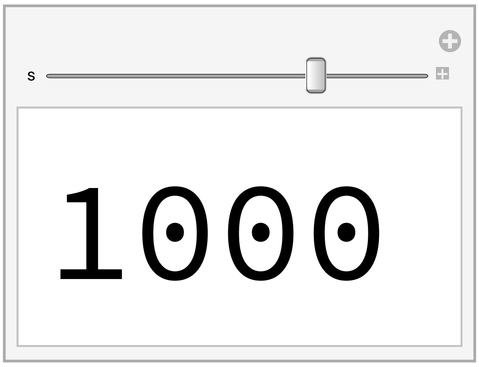


+9.2 Make a Manipulate to display 1000 at sizes between 5 and 100.

In[147]:=

```
Manipulate[Style[1000, s], {s, 5, 100}]
```

Out[147]=

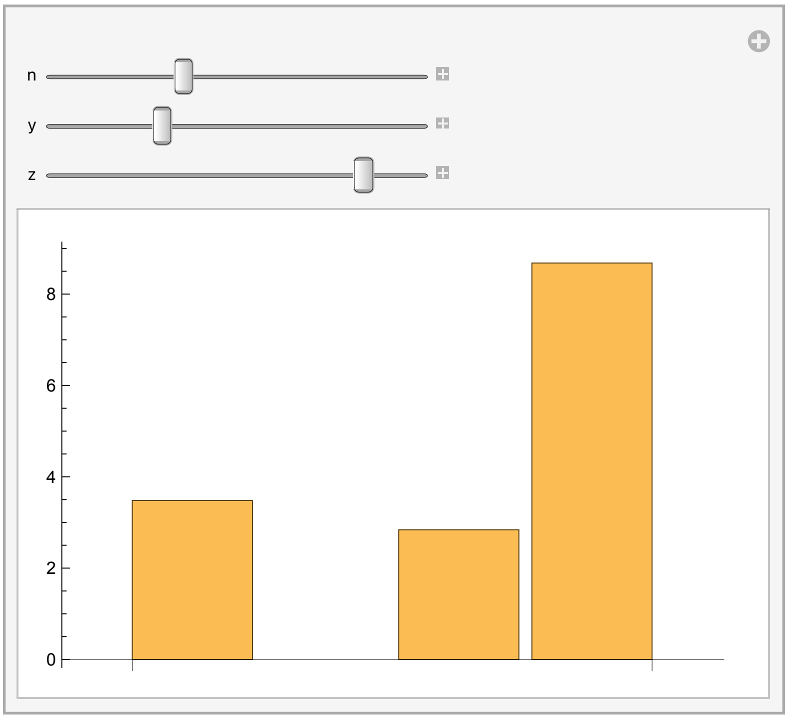


1000

+9.3 Make a Manipulate to show a bar chart with 4 bars, each with a height that can be between 0 and 10.

In[148]:=
```
Manipulate[BarChart[{n, x, y, z}], {n, 0, 10}, {y, 0, 10}, {z, 0, 10}]
```

Out[148]=



Exercises for Section 10 | Images

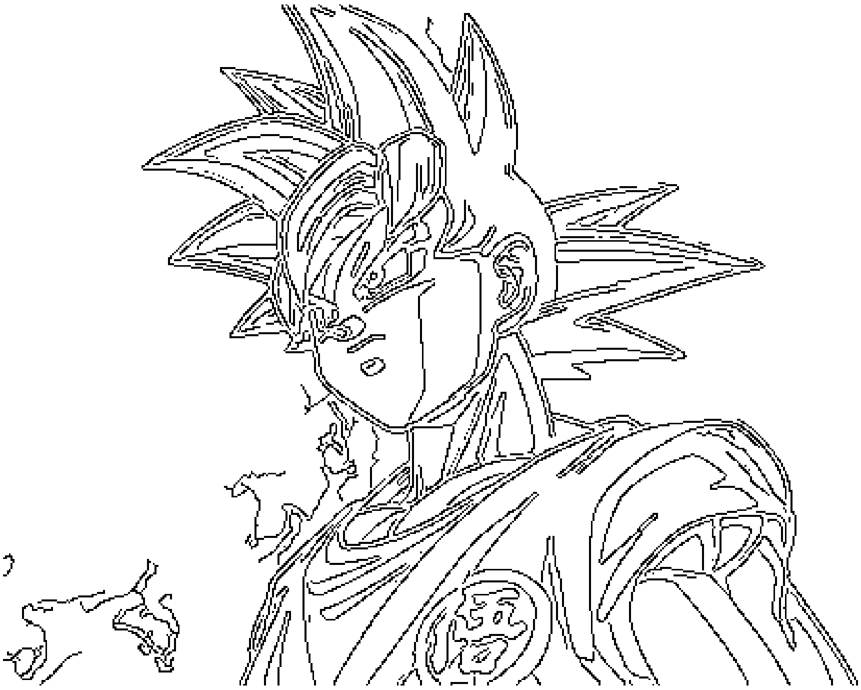
10.1 Color negate the result of edge detecting and image . (Use CurrentImage[] or any other image).

```
In[151]:= img =
```

Out[151]=

```
In[152]:= ColorNegate[EdgeDetect[img]]
```

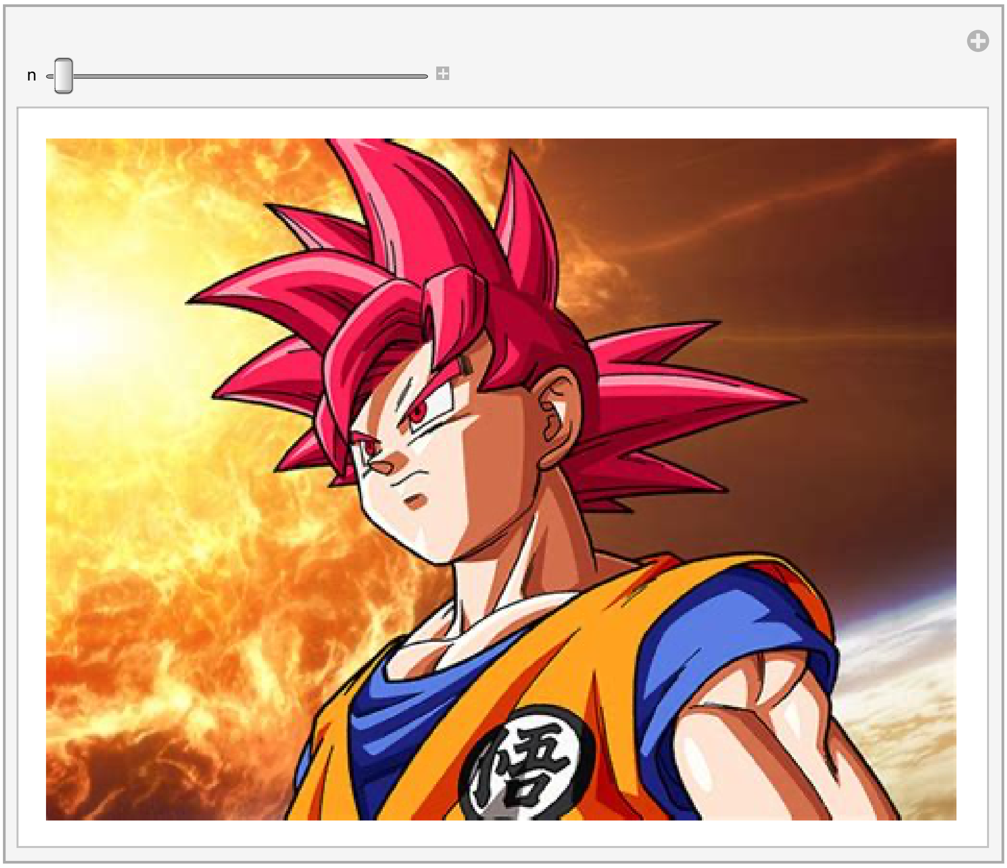
Out[152]=



10.2 Use Manipulate to make an interface for blurring and image from 0 to 20.

```
In[153]:= Manipulate[Blur[img, n], {n, 0, 20}]
```

Out[153]=



10.3 Make a table of the results from edge detecting an image with blurring from 1 to 10.

```
In[154]:= Table[Blur[EdgeDetect[img], n], {n, 1, 10}]
```

Out[154]= {

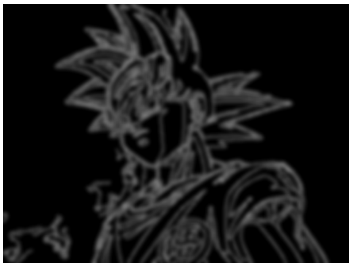
,

,

,

,

,



,

,

,

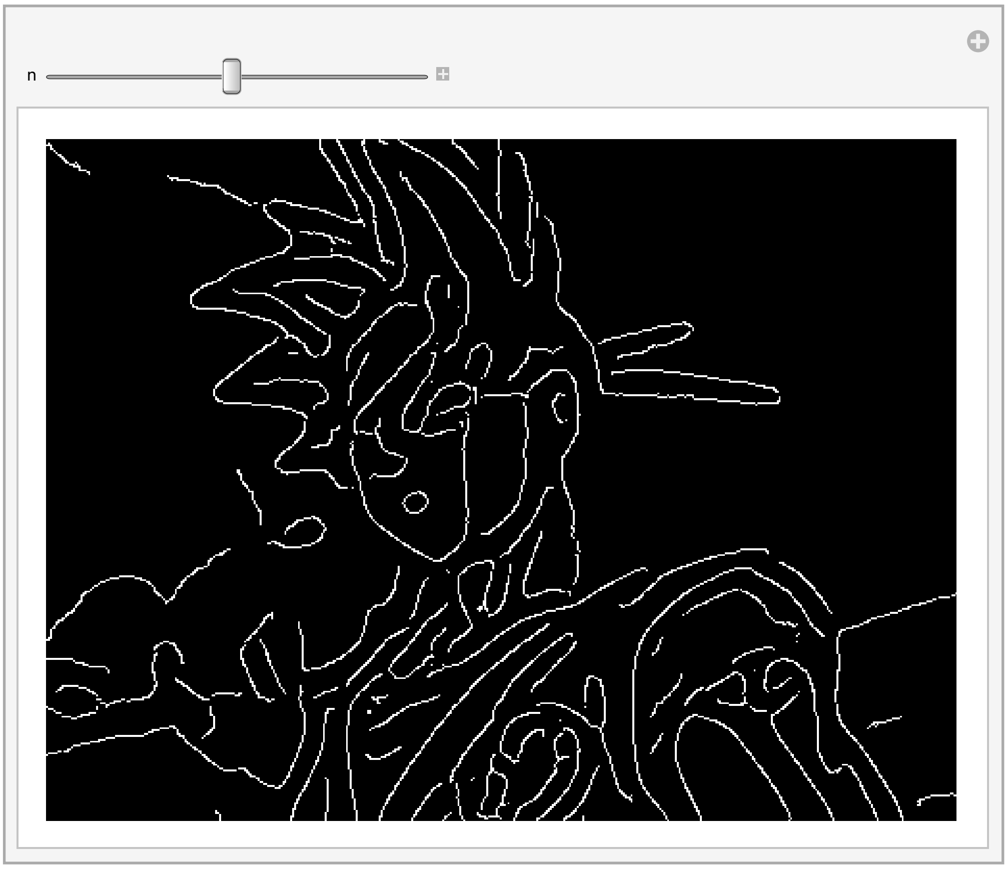
,

}

10.6 Create a Manipulate to display edges of an image as it gets blurred from 0 to 20.

```
In[155]:= Manipulate[EdgeDetect[Blur[img, n]], {n, 0, 20}]
```

Out[155]=



Exercises for Section 11 | Strings and Text

11.1 Join two copies of the string "Hello"

```
In[156]:= StringJoin["Hello", "Hello"]
Out[156]= HelloHello
```

11.2 Make a single string of the whole alphabet, in uppercase.

```
In[157]:= StringJoin[ToUpperCase[Alphabet[]]]
Out[157]= ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

11.3 Generate a string of the alphabet in reverse order.

```
In[158]:= StringReverse[StringJoin[Alphabet[]]]
Out[158]= zyxwvutsrqponmlkjihgfedcba
```

11.4 Join 100 copies of the string "AGCT".

```
In[159]:=  StringJoin[ConstantArray["AGCT", 100]]

Out[159]=  AGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAG
             CTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTA
             GCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCT
             AGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGC
             TAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAGCTAG
             CTAGCTAGCTAGCTAGCTAGCT
```

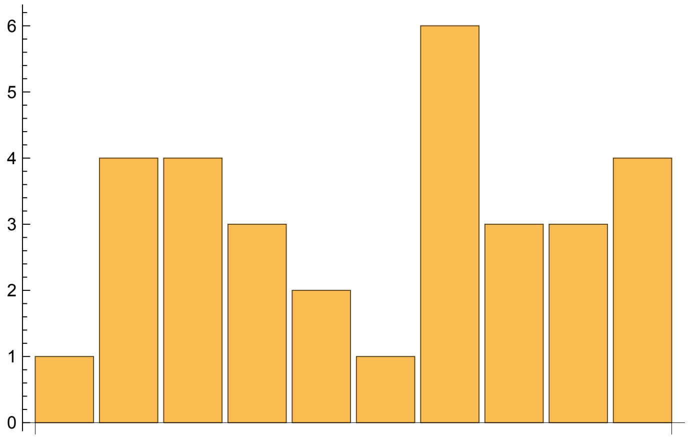
11.5 Use StringTake, StringJoin and Alphabet to get “abcdef” .

```
In[160]:=  StringTake[StringJoin[Alphabet[]], 6]

Out[160]=  abcdef
```

11.7 Make a bar chart of the lengths of the words in "A long time ago, in a galaxy far, far away".

```
In[161]:=  BarChart[
             StringLength[TextWords["A long time ago, in a galaxy far, far away."]]]
```

Out[161]=



11.8 Find the string length of the Wikipedia article for "computer".

```
In[162]:=  StringLength[WikipediaData["computer"]]

Out[162]=  57 696
```

11.9 Find how many words are in the Wikipedia article for “computer”.

```
In[166]:=  Length[TextWords[WikipediaData["computer"]]]

Out[166]=  8892
```

11.10 Find the first sentence in the Wikipedia article about “strings”.

```
In[168]:= First[TextSentences[WikipediaData["strings"]]]
Out[168]= String or strings may refer to:
```

11.11 Make a string from the first letters of all sentences in the Wikipedia article about computers.

```
In[170]:= StringJoin[StringTake[TextSentences[WikipediaData["computer"]], 1]]
Out[170]= AMTATASCESEMTTTCTPP=ATTDBTTT==DTLTTTITIIMTTAAATTTIASIBIAITITIITS=CCATFTTTAEBNH=
            DHTTTTAB==BTDETTITIRT=PTEITDTTHACIINCTLOTIIIHBT==TTHTVTE=ECWATIHJTIIAAGTBAIT=
            TJFCJTHATHTTIWITT=TTDTKIHKNNHPNIMTGFTWISTIS=TTLTTTT=C=A=SH=TC==ATIET=WTTSC=
            TSC=TCATRDIRTPIWJSAIT=TES=TTSHTALTSG=AETTLSIETWOAMTTRACrRIIISFIIG=IDOHCIAMA=
            WTOBItSTBSIT=SMSTSS=SSCIW=T=TTMIAL=TITHTFMPWSTCBOTOI=ITTSTTITMWITC=PUTTS=MF=
            ATHHIT=PALTP=ETBOBSA=CTITTITCIIA"=AWA=TMH=TQCVSLTTT=ACARPE=AT====M
```

11.12 Find the maximum word length among English words from WordList[].

```
In[171]:= Max[StringLength[WordList[]]]
Out[171]= 23
```

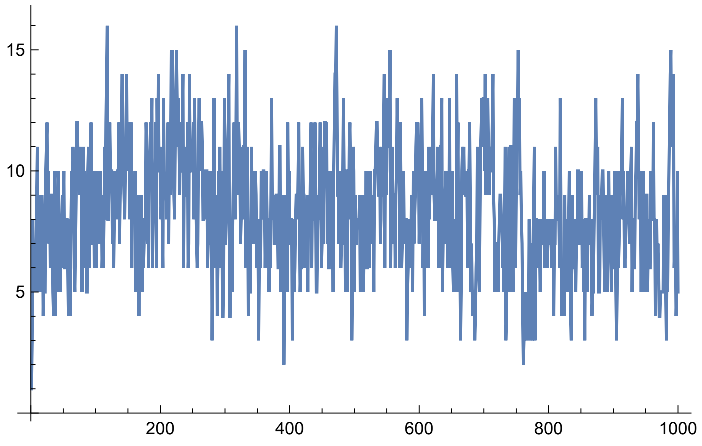
11.13 Count the number of words in WordList[] that start with “q”.

```
In[172]:= Count[StringTake[WordList[], 1], "q"]
Out[172]= 194
```

11.14 Make a line plot of the lengths of the first 1000 words from WordList[]

```
In[173]:= ListLinePlot[StringLength[Take[WordList[], 1000]]]
```

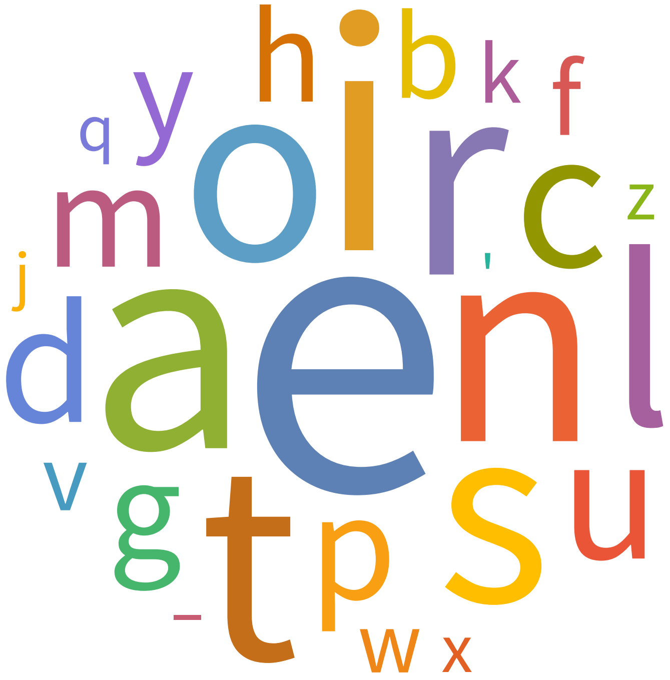
Out[173]=



11.15 Use StringJoin and Characters to make a word cloud of all letters in the words from WordList[].

```
In[174]:= WordCloud[Characters[StringJoin[WordList[]]]]
```

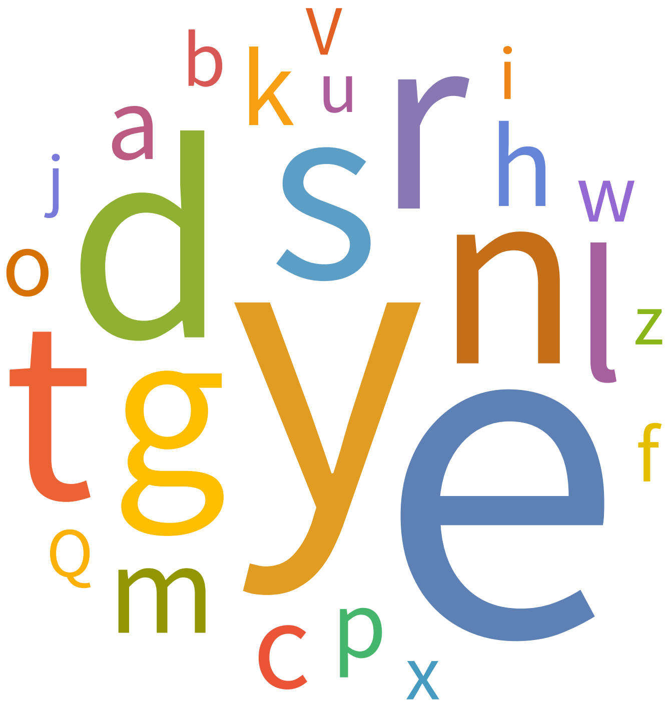
Out[174]=



11.16 Use StringReverse to make a word cloud of the last letters in the words from WordList[].

```
In[176]:= WordCloud[StringTake[StringReverse[WordList[]], 1]]
```

Out[176]=



11.17 Find the Roman numerals for the year 1959.

```
In[177]:= RomanNumeral[1959]
Out[177]= MCMLIX
```

11.18 Find the maximum string length of any Roman-numeral year from 1 to 2020.

```
In[178]:= Max[StringLength[RomanNumeral[Range[2020]]]]
Out[178]= 13
```

11.19 Make a word cloud from the first characters of the Roman numerals up to 100.

```
In[179]:= WordClod[StringTake[RomanNumeral[Range[100]], 1]]
Out[179]= WordClod[{I, I, I, I, V, V, V, V, I, X, X, X, X, X, X, X, X, X, X, X, X, X, X,
            X, X, X, X, X, X, X, X, X, X, X, X, X, X, X, X, X, X, X, X, X, X, X, X, X, X,
            L, L, L, L, L, L, L, L, L, L, L, L, L, L, L, L, L, L, L, L, L, L, L, L, L, L,
            L, L, L, L, L, L, L, L, L, L, L, L, L, L, X, X, X, X, X, X, X, X, X, X, C}]
```

11.20 Use Length to find the length of the Russian alphabet.

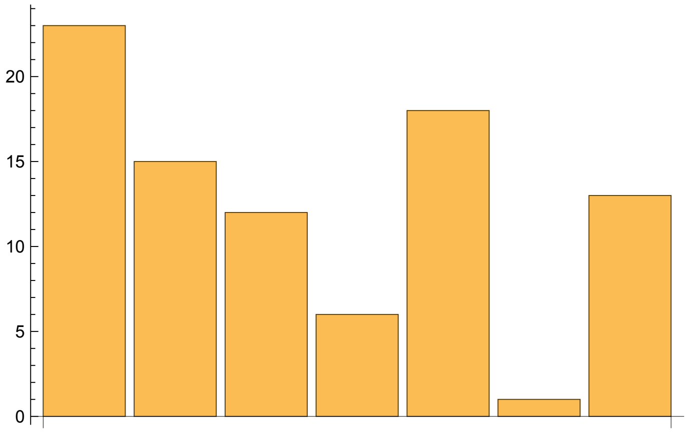
```
In[180]:= Length[Alphabet["Russian"]]
Out[180]= 33
```

11.21 Generate the uppercase Greek alphabet.

```
In[181]:= ToUpperCase[Alphabet["Greek"]]
Out[181]= {Α, Β, Γ, Δ, Ε, Ζ, Η, Θ, Ι, Κ, Λ, Μ, Ν, Ξ, Ο, Π, Ρ, Σ, Τ, Υ, Φ, Χ, Ψ, Ω}
```

11.22 Make a bar chart of the letter numbers in “wolfram”

```
In[182]:= BarChart[LetterNumber["wolfram"]]
Out[182]=
```



11.23 Use FromLetterNumber to make a string of 1000 random letters.

```
In[183]:= StringJoin[FromLetterNumber[Table[RandomInteger[25] + 1, 1000]]]

Out[183]= xqyundcqtfebfucsaghlafrzxkueptkkcvjwmhqsnqyitnilzjaijyvkbtnqfkyecmivkttslvlnjg
    suaoidqhiduastrdhrgfxacgsvjnpjmlmhzbkbqccdxyhiarfbupymjllqlplrkxznunypvrpea
    jvyjtwqmtrkwzrpegpahajepfgebbzsxhglmojtuokvnwcforgqhxmqdqiwhjuqqpfxlcqzevbi
    ezvhmkfanbjjcfzfwyhuigjnezdwxwgulmwlthilccupjztajmtklgicdeecddabpsowxxgdufr
    fwjstbqkfrogkfmoqisxxlnwsyukowmwmmqjcyjthylvyahxhvwufnrvvjmhtqvgvkddjzkjnui
    apbhbqrjzrelrbcubcvanxcbgklnlxmqrdsehkuauuztfsxoftkvuxlcfjjgjivdqhzlnlprlow
    xxponrdiarkyhbxjtnnbpgngzbeizvmwzrxnltyxydqicfzqgahrhhjeaeynjahlwrcuostyzli
    ongqkunkallrczsfbuzsodcxntpwmvyiexujpldmluwlcdgksosmboryhtdipcqnycbbttrogxl
    ouzsdynibgujshijptfimtxqveuxrmrixgywvziwwxgfbntmjvwjfjczusfrmerlterarzoggwm
    kknrgprjqsmkwocatqfwkhvutrcuchhpkmthhpdqplfrsqmdxcrzjrdfafsasfmziitcdsbcsvb
    xozudofmhxnqxytetmipzeuzxvvylrjczsczcbczpkzpukiklhyvpklrlucqdbhodlcqqbwvhgg
    izgkjumgagzvkahzjwkraqxdxcshcwjeharpkptixjoxbxeduylryriqssqzduwmxgalhzrwytv
    lnxcstpojwkuilybdrkkiuomfswxepxkpqfzzlqkxcpolgitaqfxpgrvlsjosxplsmkmzvvvful
    eewclpbirslgqfxczouqkq
```

11.24 Make a list of 100 random 5-letter stirngs.

```
In[184]:= Table[StringJoin[Table[FromLetterNumber[RandomInteger[25] + 1], 5]], 100]

Out[184]= {dxlhm, scqvc, cptot, eulwu, qlrua, dftym, ccskn, zphsm, tsqns, dxnpj, iijpc, cnemu,
  zexei, kdnxo, qbsgl, mmdyy, rbnqw, dirkv, vtsaa, iteli, qygth, ptvuo, bhsms,
  igxkz, tsnet, shwul, zodul, wovdl, iyoqp, yihzt, thrwr, bghqa, tbznq, buyyf,
  vfuwj, rwzrp, egbhr, ssxzl, vaslb, vqbtr, bdrcc, fgmgw, knhhg, dxgxl, cthbd,
  nefwx, tvbav, qnrgn, fkaga, hmuym, swtdt, emjiy, bcwen, motdd, mdlvr, erkvw,
  hqloz, qfebg, pzujl, sqmlg, jsqdd, mzxmy, rfnbh, rhekl, vimjh, iqhdw, hcnvg,
  iyens, luxne, zcusk, agfvi, xufcs, pdukm, vztfq, zjemu, vjhzw, flxyk, wnfrt,
  hxctu, rucyz, rjvny, okksl, uuuen, khqdt, eaexm, hkdia, xfrps, coedd, hlexb,
  tnkdb, wniit, bjbkl, agmrb, aclrk, pspdb, orgdg, nojzt, ddxdc, ltlhw, zmoxg}
```

11.25 Transliterate “wolfram” into Greek.

```
In[185]:= Transliterate["wolfram", "Greek"]

Out[185]= βολφραμ
```

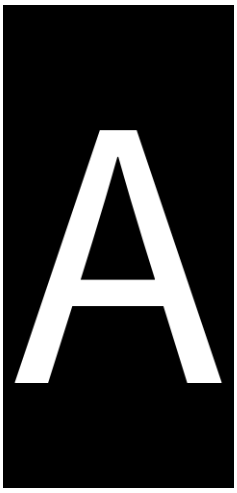
11.26 Get the Arabic alphabet and transliterate it into English.

```
In[186]:= Transliterate[Alphabet["Arabic"]]

Out[186]= {a, b, t, th, j, h, kh, d, dh, r, z, s, sh, s, d, t, z, ʿ, gh, f, q, k, l, m, n, h, w, y}
```

11.27 Make a white-on-black size-200 letter “A”.

In[187]:=
```
ColorNegate[Rasterize[Style["A", 200]]]
```

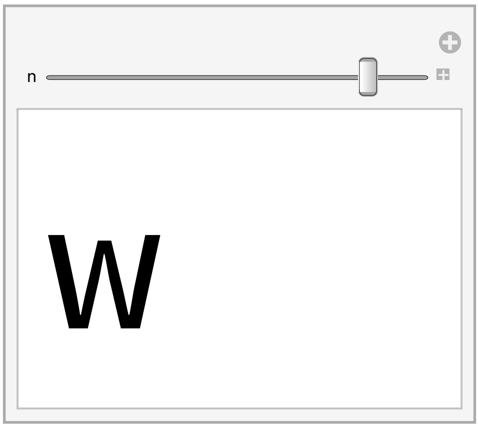
Out[187]=



11.28 Use Manipulate to make an interactive selector of size-100 characters from the alphabet, controlled by a slider.

In[188]:=
```
Manipulate[Style[FromLetterNumber[n], 100], {n, 1, Length[Alphabet[]], 1}]
```

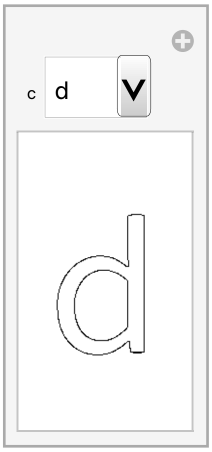
Out[188]=



11.29 Use Manipulate to make an interactive selector of black-on-white outlines of rasterized size-100 characters from the alphabet, controlled by a menu.

In[191]:=
```
Manipulate[ColorNegate[EdgeDetect[Rasterize[Style[c, 100]]]], {c, Alphabet[]}]
```

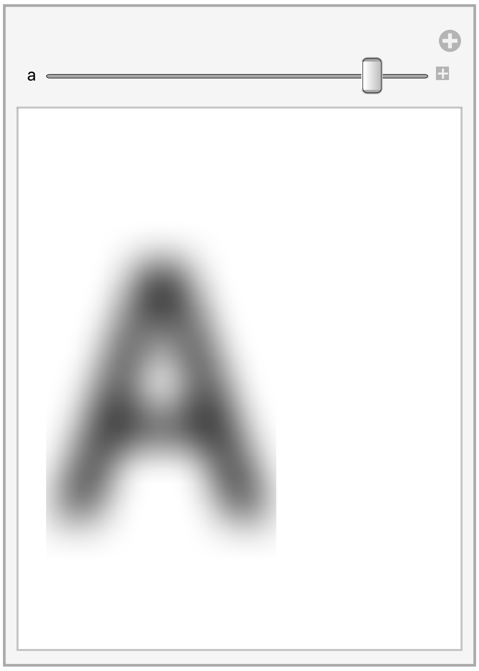
Out[191]=



11.30 Use Manipulate to create a “vision simulator” that blurs a size-200 letter “A” by an amount from 0 to 50.

In[192]:=
```
Manipulate[Blur[Rasterize[Style["A", 200]], a], {a, 0, 50}]
```

Out[192]=



+11.1 Generate a string of the alphabet followed by the alphabet written in reverse.

In[193]:=
```
StringJoin[StringJoin[Alphabet[]], StringReverse[StringJoin[Alphabet[]]]]
```

Out[193]=
```
abcdefghijklmnopqrstuvwxyzzyxwvutsrqponmlkjihgfedcba
```

+11.3 Find how many sentences are in the Wikipedia article for “computer”.

```
In[194]:= Length[TextSentences[WikipediaData["computer"]]]

Out[194]= 449
```

+11.4 Join together without spaces, etc. the words in the first sentence in the Wikipedia article for “strings”

```
In[195]:= StringJoin[TextWords[First[TextSentences[WikipediaData["strings"]]]]]

Out[195]= Stringorstringsmayreferto
```

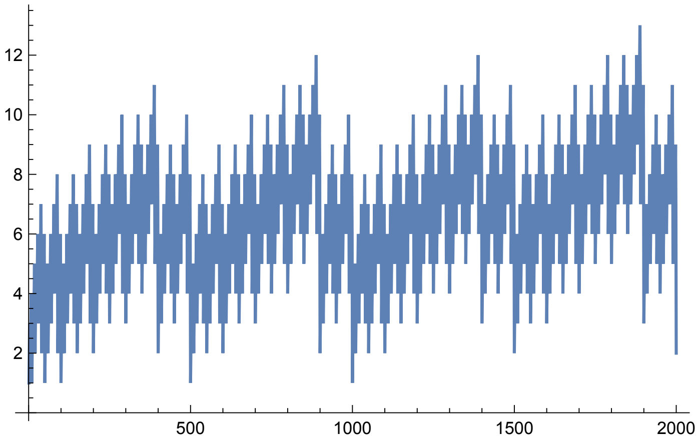
+11.5 Find the length of the longest word in the Wikipedia article about computers.

```
In[196]:= Max[StringLength[TextWords[WikipediaData["computers"]]]]

Out[196]= 25
```

+11.6 Plot the lengths of Roman numerals for numbers up to 2000.

```
In[197]:= ListLinePlot[StringLength[RomanNumeral[Range[2000]]]]
```

Out[197]=



+11.7 Generate a string by joining the Roman numerals up to 100.

```
In[198]:= StringJoin[RomanNumeral[Range[100]]]

Out[198]= IIIIIIIVVVIVIIVIIIIXXXIXIIXIIIXIVXVXVIXVIIXVIIIXIXXXXXIXXIIXXIIIXXIVXXVXXVIXXV⋱
            IIXXVIIIXXIXXXXXXXIXXXIIXXXIIIXXXIVXXXVXXXVIXXXVIIXXXVIIIXXXIXXLXLIXLIIXLII⋱
            IXLIVXLVXLVIXLVIIXLVIIIXLIXLLILIILIIILIVLVLVILVIILVIIILIXLXLXILXIILXIIILXIV⋱
            LXVLXVILXVIILXVIIILXIXLXXLXXILXXIILXXIIILXXIVLXXVLXXVILXXVIILXXVIIILXXIXLXX⋱
            XLXXXILXXXIILXXXIIILXXXIVLXXXVLXXXVILXXXVIILXXXVIIILXXXIXXCXCIXCIIXCIIIXCIV⋱
            XCVXCVIXCVIIXCVIIIXCIXC
```

+11.9 Find the maximum string length of the name of any integer up to 1000.

```
In[199]:= Max[StringLength[IntegerName[Range[1000]]]]

Out[199]= 27
```

+11.10 Make a list of uppercase size-20 letters of the alphabet in random colors.

```
In[201]:= Style[ToUpperCase[Alphabet[]], 20, RandomColor[]]

Out[201]= {A, B, C, D, E, F, G, H, I, J, K, L,
           M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z}
```

+11.11 Make a list of 100 random 5-letter strings with the Russian alphabet.

```
In[202]:= Table[
           StringJoin[Table[FromLetterNumber[RandomInteger[25] + 1, "Russian"], 5]], 100]

Out[202]= {цшнох, лбпоё, ёлжно, йохпе, лфзёц, оечдг, жпзир, ёопгр, жалиб, бцдсг, снкбн, бавцх,
           ушкхё, вкубт, фйане, ацмйм, сфшзк, дхуйш, фмрйа, хфкто, пдикм, тлшбу, рблёс,
           ебихч, ёвжех, кёпое, вкйгл, гншгп, лзхвц, рбрпз, уддда, шдбиу, ииаёц, бееум,
           дзцум, мфрчф, винсж, швнаш, пижгё, дмвог, метех, кйшоу, цзкеп, днфри, лйагж,
           чфйгб, лбкнё, дпрсф, гуофн, ииаро, лорпн, анкзв, клдут, ехедй, ётзшп, дмзмп,
           кнлхт, зфлйж, цёхгё, хггфч, зтфбк, пштчх, оезчш, тесчж, пйбхз, мштно, хдвзб,
           фиуйм, иоужч, жчупц, вркзж, вцчнд, чёзвп, шмткч, всфшд, зфхтд, емжзё, тйшбб,
           уфжап, цучач, нфмжс, хчусг, ччнбн, нцчшр, цжатц, йёццу, хеевх, ёвклг, дшопж,
           вшнсх, ёйбжк, рёрсг, алдлл, утзшр, мпссг, леффе, цфчжт, увнлз, гуйфл, мврйе}
```