This is overall very nice work. However, note that on 40.6, 40.7, and 40.8 he was looking for you to apply the new technique of patterns rather than using If[] (or Select[] or Which[]). They all work, but a lot of Section 39 was about combining patterns with function definitions. 7 1/2 / 8

PS 17 — Rania 4.8.2025

Section 39

= vs. := (delayed)

x-> vs. x :>

In[68]:=
```
(*39.1 Replace x in {x,x+1,x+2,x^2} by the same random integer up to 100. *)
{x, x + 1, x + 2, x ^ 2} /. x → RandomInteger[100]
```

Out[68]=
```
{43, 44, 45, 1849}
```

In[69]:=
```
(*39.2 Replace each x in {x,x+1,x+2,x^2}
 by a separately chosen random integer up to 100.*)
```

In[70]:=
```
{x, x + 1, x + 2, x ^ 2} /. x :→ RandomInteger[100]
```

Out[70]=
```
{42, 34, 93, 64}
```

Section 40

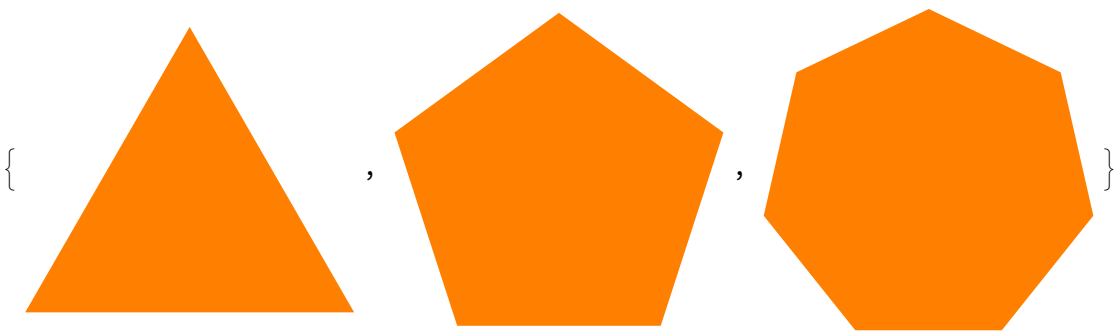
In[71]:=
```
(*40.1 Define a function f that computes the square of its argument*)
```

In[72]:=
```
f[x_] := x * x
{f[2], f[4], f[6]}
```

Out[73]=
```
{4, 16, 36}
```

In[74]:=
```
(*40.2 Define a function poly that takes an integer,
and makes a picture of an orange regular polygon with that number of sides*)
poly[n_] := Graphics[Style[RegularPolygon[n], Orange]]
{poly[3], poly[5], poly[7]}
```

Out[75]=



In[76]:=
```
(*40.3 Define a function f that takes a list
 of two elements and puts them in reverse order*)
f[{x_, y_}] := {y, x}
f[{1, 2}]
```

Out[77]=
$$\{2, 1\}$$

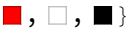
In[78]:=
```
(*40.4 Create a function f that takes two arguments and gives
 the result of multiplying them and dividing by their sum.*)
f[{x_, y_}] := x * y / (x + y)
f[{4, 5}]
```

Out[79]=
$$\frac{20}{9}$$

In[80]:=
```
(*40.5 Define a function f that takes a list of two
 elements and returns a list of their sum,difference and ratio *)
f[{x_, y_}] := {x + y, x - y, x / y}
f[{3, 4}]
```

Out[81]=
$$\left\{7, -1, \frac{3}{4}\right\}$$

In[82]:=
```
(*40.6 Define a function evenodd that gives Black if its argument
 is even and White otherwise,but gives Red if its argument is 0 *)
evenodd[x_] := If[x == 0, Red, If[EvenQ[x] == True, Black, White]]
{evenodd[0], evenodd[1], evenodd[2]} (*lol the egyptian flag colors*)
```

Out[83]=


In[84]:=
```
(*40.7 Define a function f of three arguments where
     the second two arguments are added if the first argument is 1,
    multiplied if it's 2 and raised to a power if it's 3*)
f[{x_, y_, z_}] :=
 If[x == 1, y + z, If[x == 2, y * z, If[x == 3, y^z, "First Argument is not 1, 2, 3"]]]
{f[{1, 2, 3}], f[{2, 2, 3}], f[{3, 2, 3}]}
```

Out[85]=
$$\{5, 6, 8\}$$

In[86]:=
```
(*40.8Define a Fibonacci function f with f[0] and f[1] both being 1,
and f[n] for integer n being the sum of f[n-1] and f[n-2]*)
f[x_] := If[x ≠ 0 && x ≠ 1, f[x - 1] + f[x - 2], 1]
f /@ Range[5]
```

Out[87]=
$$\{1, 2, 3, 5, 8\}$$

In[88]:=
```
(*40.9 Create a function animal that takes a string,
and gives a picture of an animal with that name.*)
animal[x_] := Interpreter["Animal"][x][image]
animal[#] & /@ {"horse", "cat", "camel"}
```

Out[89]=

In[90]:=
```
(*40.10 Define a function nearwords that takes a string and an integer n,
and gives the n words in WordList[] that are nearest to a given string.*)
```

In[91]:=
```
nearwords[x_, n_] := Nearest[WordList[], x, n + 1]
nearwords["hello", 2]
nearwords["brian", 9]
nearwords["waves", 22]
```

Out[92]=
```
{hello, cello}
```

Out[93]=
```
{bran, aria, avian, bairn, ban, barman, bean, bias, bin}
```

Out[94]=
```
{eaves, wages, wave, waver, aver, cave, elves, fives, gavel, have, haven,
 heaves, hives, lave, mates, maven, names, nave, navel, pave, paved, rates}
```