that the loop will "spin" at most 30 times per second. The `update` command will insert an appropriate pause each time through to maintain a relatively constant rate. Of course, the rate throttling will only work when the body of the loop itself executes in less than 1/30th of a second.

Example: 1000 frames at 30 frames per second

```
win = GraphWin("Update Example", 320, 200, autoflush=False)
for i in range(1000):
    # <drawing commands for ith frame>
    update(30)
```

4.9 Chapter Summary

This chapter introduced computer graphics and object-based programming. Here is a summary of some of the important concepts:

- An object is a computational entity that combines data and operations. Objects know stuff and can do stuff. An object's data is stored in instance variables, and its operations are called methods.
- Every object is an instance of some class. It is the class that determines what methods an object will have. An instance is created by calling a constructor method.
- An object's attributes are accessed via dot notation. Generally computations with objects are performed by calling on an object's methods. Accessor methods return information about the instance variables of an object. Mutator methods change the value(s) of instance variables.
- The graphics module supplied with this book provides a number of classes that are useful for graphics programming. A `GraphWin` is an object that represents a window on the screen for displaying graphics. Various graphical objects such as `Point`, `Line`, `Circle`, `Rectangle`, `Oval`, `Polygon`, and `Text` may be drawn in a `GraphWin`. Users may interact with a `GraphWin` by clicking the mouse or typing into an `Entry` box.
- An important consideration in graphical programming is the choice of an appropriate coordinate system. The graphics library provides a way of automating certain coordinate transformations.

- The situation where two variables refer to the same object is called aliasing. Aliasing can sometimes cause unexpected results. Use of the `clone` method in the graphics library can help prevent these situations.

4.10 Exercises

Review Questions

True/False

1. Using `graphics.py` allows graphics to be drawn in a Python shell window.
2. Traditionally, the upper-left corner of a graphics window has coordinates (0,0).
3. A single point on a graphics screen is called a pixel.
4. A function that creates a new instance of a class is called an accessor.
5. Instance variables are used to store data inside an object.
6. The statement `myShape.move(10,20)` moves `myShape` to the point (10,20).
7. Aliasing occurs when two variables refer to the same object.
8. The `copy` method is provided to make a copy of a graphics object.
9. A graphics window always has the title "Graphics Window."
10. The method in the graphics library used to get a mouse click is `readMouse`.

Multiple Choice

1. A method that returns the value of an object's instance variable is called a(n)
   a) mutator b) function c) constructor d) accessor
2. A method that changes the state of an object is called a(n)
   a) stator b) mutator c) constructor d) changor
3. What graphics class would be best for drawing a square?
   a) `Square` b) `Polygon` c) `Line` d) `Rectangle`

4. What command would set the coordinates of `win` to go from (0,0) in the lower-left corner to (10,10) in the upper-right?
   a) `win.setcoords(Point(0,0), Point(10,10))`
   b) `win.setcoords((0,0), (10,10))`
   c) `win.setcoords(0, 0, 10, 10)`
   d) `win.setcoords(Point(10,10), Point(0,0))`

5. What expression would create a line from (2,3) to (4,5)?
   a) `Line(2, 3, 4, 5)`
   b) `Line((2,3), (4,5))`
   c) `Line(2, 4, 3, 5)`
   d) `Line(Point(2,3), Point(4,5))`

6. What command would be used to draw the graphics object `shape` into the graphics window `win`?
   a) `win.draw(shape)` b) `win.show(shape)`
   c) `shape.draw()` d) `shape.draw(win)`

7. Which of the following computes the horizontal distance between points `p1` and `p2`?
   a) `abs(p1-p2)`
   b) `p2.getX() - p1.getX()`
   c) `abs(p1.getY() - p2.getY())`
   d) `abs(p1.getX() - p2.getX())`

8. What kind of object can be used to get text input in a graphics window?
   a) `Text` b) `Entry` c) `Input` d) `Keyboard`

9. A user interface organized around visual elements and user actions is called a(n)
   a) GUI b) application c) windower d) API

10. What color is `color_rgb(0,255,255)`?
    a) yellow b) cyan c) magenta d) orange

Discussion

1. Pick an example of an interesting real-world object and describe it as a programming object by listing its data (attributes, what it "knows") and its methods (behaviors, what it can "do").

2. Describe in your own words the object produced by each of the following operations from the graphics module. Be as precise as you can. Be sure to mention such things as the size, position, and appearance of the various objects. You may include a sketch if that helps.

a)
```
Point(130,130)
```

b)
```
c = Circle(Point(30,40),25)
c.setFill("blue")
c.setOutline("red")
```

c)
```
r = Rectangle(Point(20,20), Point(40,40))
r.setFill(color_rgb(0,255,150))
r.setWidth(3)
```

d)
```
l = Line(Point(100,100), Point(100,200))
l.setOutline("red4")
l.setArrow("first")
```

e)
```
Oval(Point(50,50), Point(60,100))
```

f)
```
shape = Polygon(Point(5,5), Point(10,10), Point(5,10), Point(10,5))
shape.setFill("orange")
```

g)
```
t = Text(Point(100,100), "Hello World!")
t.setFace("courier")
t.setSize(16)
t.setStyle("italic")
```

3. Describe what happens when the following interactive graphics program runs:

```
from graphics import *

def main():
    win = GraphWin()
    shape = Circle(Point(50,50), 20)
    shape.setOutline("red")
    shape.setFill("red")
    shape.draw(win)
    for i in range(10):
        p = win.getMouse()
        c = shape.getCenter()
        dx = p.getX() - c.getX()
```

```
            dy = p.getY() - c.getY()
            shape.move(dx,dy)
        win.close()
main()
```

Programming Exercises

1. Alter the program from the last discussion question in the following ways:

   (a) Make it draw squares instead of circles.

   (b) Have each successive click draw an additional square on the screen (rather than moving the existing one).

   (c) Print a message on the window "Click again to quit" after the loop, and wait for a final click before closing the window.

2. An archery target consists of a central circle of yellow surrounded by concentric rings of red, blue, black and white. Each ring has the same width, which is the same as the radius of the yellow circle. Write a program that draws such a target. *Hint*: Objects drawn later will appear on top of objects drawn earlier.

3. Write a program that draws some sort of face.

4. Write a program that draws a winter scene with a Christmas tree and a snowman.

5. Write a program that draws 5 dice on the screen depicting a straight (1, 2, 3, 4, 5 or 2, 3, 4, 5, 6).

6. Modify the graphical future value program so that the input (principal and APR) also are done in a graphical fashion using `Entry` objects.

7. Circle Intersection.

Write a program that computes the intersection of a circle with a horizontal line and displays the information textually and graphically.

Input: Radius of the circle and the y-intercept of the line.

Output: Draw a circle centered at $(0, 0)$ with the given radius in a window with coordinates running from -10,-10 to 10,10.
Draw a horizontal line across the window with the given y-intercept.
Draw the two points of intersection in red.
Print out the x values of the points of intersection.

Formula: $x = \pm\sqrt{r^2 - y^2}$

8. Line Segment Information.

This program allows the user to draw a line segment and then displays some graphical and textual information about the line segment.

Input: Two mouse clicks for the end points of the line segment.

Output: Draw the midpoint of the segment in cyan.
Draw the line.
Print the length and the slope of the line.

Formulas:
$$dx = x_2 - x_1$$
$$dy = y_2 - y_1$$
$$slope = dy/dx$$
$$length = \sqrt{dx^2 + dy^2}$$

9. Rectangle Information.

This program displays information about a rectangle drawn by the user.

Input: Two mouse clicks for the opposite corners of a rectangle.

Output: Draw the rectangle.
Print the perimeter and area of the rectangle.

Formulas:
$$area = (length)(width)$$
$$perimeter = 2(length + width)$$

10. Triangle Information.

Same as the previous problem, but with three clicks for the vertices of a triangle.

Formulas: For perimeter, see length from the Line Segment problem. $area = \sqrt{s(s-a)(s-b)(s-c)}$ where a, b, and c are the lengths of the sides and $s = \frac{a+b+c}{2}$.

11. Five-click House.

You are to write a program that allows the user to draw a simple house using five mouse clicks. The first two clicks will be the opposite corners of the rectangular frame of the house. The third click will indicate the center of the top edge of a rectangular door. The door should have a total width that is $\frac{1}{5}$ of the width of the house frame. The sides of the door should extend from the corners of the top down to the bottom of the frame. The fourth click will indicate the *center* of a square window. The window is half as wide as the door. The last click will indicate the peak of the roof. The edges of the roof will extend from the point at the peak to the corners of the top edge of the house frame.