Basic Movement and Draw Commands of Turtle Library

1. forward(distance) | fd(distance)
Parameter: distance – a number (integer or float)
Move the turtle forward by the specified *distance*, in the direction the turtle is headed.

2. backward(distance) | bk(distance) | back(distance)
Parameter: distance – a number (integer or float)
Move the turtle backward by *distance*, opposite to the direction the turtle is headed. Do not change the turtle's heading.

3. right(angle) | rt(angle)
Parameter: angle – a number (integer or float)
Turn turtle right by *angle* units. (Units are in degrees, by default.)

4. left(angle) | lt(angle)
Parameter: angle – a number (integer or float)
Turn turtle left by *angle* units. (Units are in degrees, by default.)

5. goto(x, y) | setpos(x, y) | setposition(x, y)
Parameters: x – a number, y – a number
Move turtle to an absolute position. If the pen is down, draw line. Do not change the turtle's orientation. [Note: this works for two numbers like this: goto(100,200) and also for one tuple of two numbers like this: goto ((100,200)).

etx(x)
Parameter: x – a number (integer or float)
Set the turtle's first coordinate to *x*, leave second coordinate unchanged.
sety(y)
Parameter: y – a number (integer or float)
Set the turtle's second coordinate to *y*, leave first coordinate unchanged.

6. setheading(angle) | seth(angle)
Parameter: angle – a number (integer or float)
Set the orientation of the turtle to *angle*. (The angle is in degrees, by default.)
If you set the heading to 0, the turtle will point to the right (East).
If you set the heading to 90, the turtle will point up (North).
If you set the heading to 180, the turtle will point to the left (West).
If you set the heading to 270, the turtle will point down (South).

7. home()
There are no parameters.
Move turtle to the origin – coordinates (0,0) – and set its heading to its initial orientation (which is pointing to the right (East)).

8. circle(radius)
Parameter: radius – a number (integer or float)
Draw a circle with the given radius. The center is radius units to the left of the turtle. (Remember that the turtle is always pointed in some direction (depending on what has happened up to the point in the program). This direction is called the turtle's heading. You can change the turtle's heading using the setheading function. The center of the circle (or other polygon) drawn by the circle function is a distance of radius from the left side of the turtle – 90 degrees to the left of the turtle's current heading.)

9. circle(radius, extent=None, steps=None)
You can also supply a second parameter: extent
If *extent* is not given, draw the entire circle. If *extent* is not a full circle, one endpoint of the arc is the current pen position. Draw the arc in counterclockwise direction if *radius* is positive, otherwise in clockwise direction. Finally the direction of the turtle is changed by the amount of *extent*.

10.stamp()
No parameters.

Stamp a copy of the turtle shape onto the canvas at the current turtle position. Return a stamp_id for that stamp, which can be used to delete it by calling clearstamp(stamp_id).

11. speed(speed = None)

Parameter: speed – an integer in the range of 0 .. 10 or a speedstring
Set the turtle's speed to an integer value in the range 0..10. If no argument is given, return current speed.
If input is a number greater than 10 or smaller than 0.5, speed is set to 0.
Speedstrings are mapped to speedvalues as follows:

- "fastest": 0
- "fast": 10
- "normal": 6
  "slow":3

Speeds from 1 to 10 enforce increasingly faster animation of line drawing and turtle turning.
Attention: *speed* = 0 means that *no* animation takes place. forward/back makes turtle jump and likewise left/right make the turtle turn instantly.
Tell Turtle's state

12. towards(x, y=None)

Parameters: x – a number (x location of a point); y – a number (y location of a point)
Return the angle between the line from turtle position to the point specified by (x,y). This works for two numbers like this: towards(50,-50) and also for a tuple of two numbers like this: towards((50,-50))

13. distance(x, y)

Parameters: x – a number (x location of a point); y – a number (y location of a point)
Return the distance from the turtle to the point (x,y). This works for two numbers like this: distance(100,100) or for a tuple of two numbers like this: distance((100,100))

Drawing State

14. pendown() | pd() | down()

No parameter.
Put the pen down. There will be drawing when you move the turtle when the pen is down.

15. penup() | pu() | up()
No parameter.
Put the pen up. There won't be any drawing when you move the turtle while the pen is up.

16. pensize(width) | width(width)
Parameter: width – a positive number
If you don't supply a parameter, this function will return the current width of the pen. If you supply a parameter, this will set the line thickness to the number you supply.

17. pencolor()
Parameter: either no parameter or a string representing a color or three numbers (inside parentheses) representing a color.
If you don't supply a parameter, this function will return the current color of the pen. If you supply a parameter (either a string representing a color or three numbers representing a color), it will set the pen color to the color you provided.

18. fillcolor()
Parameter: either no parameter or a string representing a color or three numbers (inside parentheses) representing a color.
Parameter: either no parameter or a string representing a color or three numbers (inside parentheses) representing a color.
Return or set the fill color.

Filling
19. begin_fill()
No parameters.
To be called just before drawing a shape to be filled.

20. end_fill()
No parameters.
To be called just after you have drawn the shape that you want to fill.
More drawing control

21. clear()
Delete the turtle's drawings from the screen. Do not move turtle. State and position of the turtle as well as drawings of other turtles are not affected.

22. write(msg)
Parameter: msg – a text string (the message that you want to display)
Write the text – the string representation of msg – at the current turtle position aligned to the left in Arial font, size 8. The pen will not move.
You can supply more parameters like this:
write(*arg*, *move=False*, *align="left"*, *font=("Arial", 8, "normal")*
Parameters: arg – a text string (the message that you want to display)
move – either True or False depending on whether or not you want the turtle to move to the lower right corner of the text displayed
align – either "left", "center", or "right" depending on how you want the text aligned.
font – you supply the name of the font, the size of the font – a positive integer, and one of the three values "

23. showturtle() | st()
No parameters.
Make the turtle visible.

24.hideturtle() | ht()
No parameters.
Hide the turtle. (Make it invisible) You will still see the drawing – you just won't see the turtle.

Appearance
25. shape(tshape)
Parameter: tshape – a string indicating the turtle shape. The built-in shapes are "arrow", "turtle", "circle", "square", "triangle", and "classic".
If you don't provide a parameter, this function will return the current turtle shape.
If you supply one of these parameters shown, it will set the turtle shape to the shape you supply.

24. window_width()
Return the width of the turtle window.
window_height() Return the height of the turtle window.

Window control
26. bgcolor()
Parameter: a string representing a color or three numbers (in parentheses) representing a color.

Set or return background color of the TurtleScreen. If you don't provide a parameter, this function will return the background color. If you do provide a parameter, it will set the parameter to the color you provide.

27. screensize(canvwidth=None, canvheight=None, bg=None)
Parameters: canvwidth – positive integer, new width of the canvas in pixels
canvheight – positive integer, new height of the canvas in pixels
bg – text string representing a color or three numbers (in parentheses) representing a color
If no parameters are supplied, this function returns the canvas width, the canvas height, and the background color. If any of these parameters are supplied, then it sets the canvas height, or canvas width, or background color to the parameter given.

28. colormode(cmode=None)
Parameters: cmode – either the value 1.0 or the value 255
If you don't supply a parameter, it returns the current color mode (which is either 1.0 or 255).
If you supply the parameter value of 1.0, you will set the color mode to 1.0.
If you supply the parameter value of 255, you will set the color mode to 255.
When you supply a color (to other functions) as three numbers, then the numbers you use must be between 0 and 1.0 (if you use a color mode of 1.0) or the numbers you use must be between 0 and 255 (if you use a color mode of 255).
29. window_height()
No parameter.
Return the height of the turtle window.
30. window_width()
No parameter.
Return the width of the turtle window.

31. textinput(title, prompt)
numinput()
Methods specific to Screen

32. exitonclick()
Shut the turtlegraphics window on mouse click.

33.setup(width, height, startx, starty) Set the size and position of the main window.

34. title(titlestring)

Библиотека `turtle` – это расширения языка Питон, позволяющее рисовать на экране несложные рисунки. Представьте себе, что по экрану компьютера ползает маленькая черепашка `(turtle)`. Вы можете управлять движением черепашки, отдавая ей различные команды вида "Проползти вперед на 10 пикселей", "Повернуть направо", "Повернуть налево". После того, как вы отдадите ей команду "Начать рисовать", черепашка будет оставлять за собой след, пока не получит команду "Кончить рисовать". Управлять черепашкой можно при помощи инструкций Питона. Вот как, например, выглядит программа, рисующая квадрат:

```
import turtle           # Подключаем модуль turtle


turtle.reset()          # Очищаем экран, приводим черепашку в начальное
положение


turtle.pendown()        # Опускаем перо перо (начало рисования)


turtle.forward(50)      # Проползти 50 пикселей вперед

turtle.left(90)         # Поворот влево на 90 градусов

turtle.forward(50)      # Рисуем вторую сторону квадрата

turtle.left(90)

turtle.forward(50)      # Рисуем третью сторону квадрата

turtle.left(90)

turtle.forward(50)      # Рисуем четвертую сторону квадрата


turtle.penup()          # Поднять перо (закончить рисовать)


turtle.forward(100)     # Отвести черепашку от рисунка в сторону
```

```
turtle.mainloop()       # Задержать окно на экране
```

Документация

Находится на странице //docs.python.org/3/library/turtle.html.

Основные команды для управления черепашкой

Ползаем

`forward(distance)`
: Проползти вперёд на distance пикселей;

`backward(distance)`
: Проползти назад на distance пикселей;

`right(angle)`
: Повернуться налево на angle градусов;

`left(angle)`
: Повернуться направо на angle градусов;

`goto(x, y)`
: Переместить черепашку в точку с координатами (x,y);

`setx(x)`
: Установить x координату черепашки;

`sety(y)`
: Установить y координату черепашки;

`setheading(to_angle)`
: Повернуть черепашку под углом to_angle к вертикали (0 — наверх, 90 — направо);

`home()`
: Вернуть черепашку домой — в точку, с координатами (0,0);

`circle(radius)`
: Нарисовать окружность радиуса |r|, центр которой находится слева от черепашки, если r>0 и справа, если r<1;

`dot(size, color)`
: Нарисовать точку диаметра size цвета color. Параметр color необязателен;

`undo()`
: Откатить предыдущее действие черепашки;

`speed(speed)`
: Установить скорость черепашки. speed должно быть от 1 (медленно) до 10 (быстро), или 0 (мгновенно);

Рисуем

`pendown()`
: Начать рисовать;

`penup()`
: Закончить рисовать;

`pensize(width)`
: Установить диаметр пера в width;

`pencolor(colorstring)`
: Установить цвет линии, которая рисует черепашка (например, `'brown'` или `'#32c18f'`);

`fillcolor(colorstring)`
: Установить цвет заполнения;

`begin_fill()`
: Начать следить за черепашкой для заполнения области;

`end_fill()`
: Заполнить цветом `fillcolor` область, пройденную черепашкой начиная с `begin_fill()`;

`showturtle()`
: Показать черепашку;

`hideturtle()`
: Спрятать черепашку;

`write(text)`
: Вывести текст `text`;

Узнать про черепашку

`position()`
: Получить текущие координаты черепашки;

`towards(x, y)`
: Получить угол между текущим направление черепашки и прямой от черепашки к точке (x,y);

`xcor()`
: Получить x координату черепашки;

`ycor()`
: Получить y координату черепашки;

`heading()`
: Получить текущий угол к вертикали;

`distance(x, y)`
: Получить расстояние до точки (x,y);

`isdown()`
: Узнать, рисует ли сейчас черепашка;

`isvisible()`
: Узнать, видима ли сейчас черепашка;

Интерактив

`onkey(function, key)`
: Выполнить функцию `function` (принимающей два аргумента, x и y — координаты черепашки) после нажатия кнопки `key` (например, 'a', 'Up', 'space');

`listen()`
: Начать следить на нажатиями клавиш и кликами мыши;

`ontimer(function, time)`
: Выполнить функцию `function` через `time` миллисекунд;

`textinput(title, prompt)`
: Вывести окно с заголовком `title` и текстом `prompt`, вернуть введённое значение;