		turtle.fill(0)
write ('строка')	Вывод текста в текущей позиции пера	turtle.write('Начало координат!')
tracer(flag)	Включение (flag=1) и выключение (flag=0) режима отображения указателя "пера" ("черепашки"). По умолчанию включён.	turtle.tracer(0)
clear()	Очистка области рисования	turtle.clear(0)

При выключенном режиме отображения указателя "черепашки" рисование происходит значительно быстрее, чем при включённом.

Нужно заметить, что хотя углы поворота исполнителя изначально интерпретируются в градусах, при использовании тригонометрических функций модуля turtle (например, turtle.sin()) аргументы этих функций воспринимаются как радианы.

Оборудование и материалы.

Персональный компьютер, среда разработки Python.

Указания по технике безопасности:

Соответствуют технике безопасности по работе с компьютерной техникой.

Задания

Проделаем упражнение с целью определить систему координат окна рисования. Приведённый ниже код формирует картинку, показанную на рис. 14.2.

```
# -*- coding: utf-8 -*-
  import turtle
#
turtle.reset()
turtle.tracer(0)
turtle.color('#0000ff')
#
turtle.write('0,0')
#
turtle.up()
x=-170
y=-120
coords=str(x)+","+str(y)
turtle.goto(x, y)
turtle.write(coords)
#
x=130
y=100
coords=str(x)+","+str(y)
turtle.goto(x, y)
turtle.write(coords)
#
x=0
y=-100
coords=str(x)+","+str(y)
turtle.goto(x, y)
turtle.write(coords)
#
turtle.down()
x=0
y=100
```

```
coords=str(x)+","+str(y)
turtle.goto(x, y)
turtle.write(coords)
#
turtle.up()
x=-150
y=0
coords=str(x)+","+str(y)
turtle.goto(x, y)
turtle.write(coords)
#
turtle.down()
x=150
y=0
coords=str(x)+","+str(y)
turtle.goto(x, y)
turtle.write(coords)
turtle.up()
#
turtle.mainloop()
```

Здесь строка с координатами формируется "в лоб", путём конкатенации преобразованных в строки значений координат.

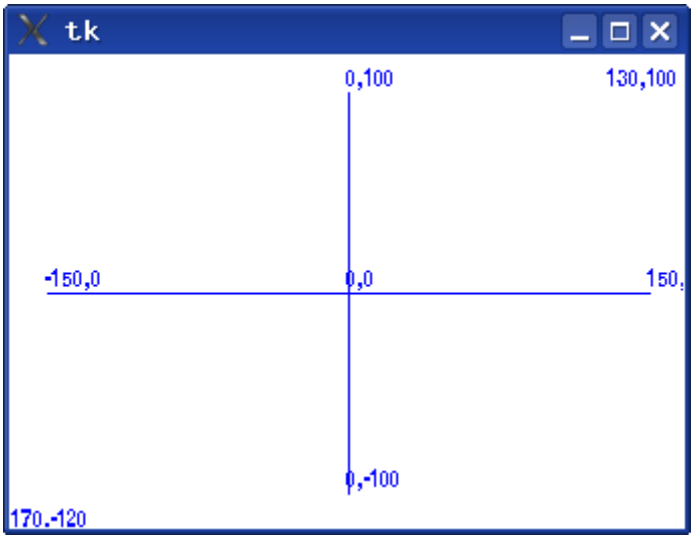

Рис. 14.2. Система координат окна рисования

Картинка, показанная на рис. 14.3, сформирована нижеследующим кодом.

```
# -*- coding: utf-8 -*-
  import turtle
#
turtle.reset()
turtle.tracer(0)
turtle.width(2)
#
turtle.up()
x=0
y=-100
turtle.goto(x, y)
turtle.begin_fill()
turtle.color('#ffaa00')
```

```
turtle.down()
turtle.circle(100)
turtle.end_fill()
turtle.color('black')
turtle.circle(100)
turtle.up()
#
x=-45
y=50
turtle.goto(x, y)
turtle.down()
turtle.color('#0000aa')
turtle.begin_fill()
turtle.circle(7)
turtle.up()
turtle.end_fill()
#
x=45
y=50
turtle.goto(x, y)
turtle.down()
turtle.color('#0000aa')
turtle.begin_fill()
turtle.circle(7)
turtle.up()
turtle.end_fill()
#
x=-55
y=-50
turtle.goto(x, y)
turtle.right(45)
turtle.width(3)
turtle.down()
turtle.color('#aa0000')
turtle.circle(80, 90)
turtle.up()
#
turtle.right(135)
x=0
y=50
turtle.goto(x, y)
turtle.width(2)
turtle.color('black')
turtle.down()
turtle.forward(100)
#
turtle.mainloop()
```

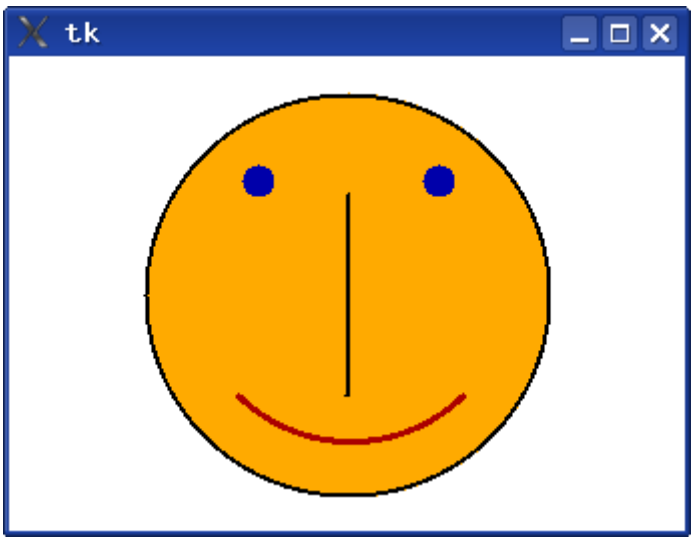

Рис. 14.3. Пример формирования изображения

Для того, чтобы изобразить улыбку, потребовалось после перемещения пера в начальную точку дуги (левую) повернуть перо на 45 градусов. Дело в том, что изначально направлением "вперёд" для пера является направление вправо (как показано на рис. 14.1). Окружности и дуги рисуются как касательные к этому "вектору", начинаясь в точке с текущими координатами пера. Поэтому для улыбки потребовалось изменить направление "вектора".

Далее, перо, первоначально сориентированное на 45 градусов вправо, после прохождения дуги в 90 градусов соответственно изменило своё направление. Поэтому для получения вертикальной линии его пришлось дополнительно повернуть.

Можно поэкспериментировать с рисованием домиков, солнышка и более сложных композиций. Однако для формирования сложных кривых (например, графиков функций) с помощью этого модуля придётся многократно выполнять команду goto(x,y). В этом легко убедиться, попытавшись нарисовать, например, график параболы.

Задания и упражнения

1. Как в примерах кода, формирующего изображения на рис. 14.2 и рис. 14.3, применить кортежи?
2. Напишите код для создания изображения "домика" (квадрат под треугольником) без подъёма пера при условии однократного перемещения по каждой линии.
3. Рассчитайте координаты и напишите код для создания изображения "солнца" (круг и расходящиеся от него отрезки) так, чтобы "лучи" начинались на расстоянии 2 точки от круга (не менее 8-ми лучей).
4. Напишите код для построения графика степенной функции ($y = ax^b$) с началом координат в левой нижней четверти окна рисования так, чтобы кривая проходила практически через всё окно.

Контрольные вопросы

1. Назначение модуля turtle.
2. Метод mainloop(), вывод окна.
3. Команды перемещения пера модуля turtle.
4. Настройка параметров пера.
5. Черчение объектов в модуле turtle.
6. Рисование в окне модуля turtle.
7. Создание графиков функций.