Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

Высшего профессионального образования

«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМ. Р. Е. АЛЕКСЕЕВА»

Кафедра «Автоматизация машиностроения»

**КУРСОВАЯ РАБОТА**

по дисциплине «Программирование и алгоритмизация»: Нормирование точности и

Тема «Разработка программы «черепашка»

Вариант 22

Выполнил:

Студент гр. 20-РТ

Порфирьева А.Н.

Проверил:

Тюриков М.И.

Нижний Новгород

2022

**Содержание**

1. Постановка задачи……………………………………………….……….3
2. Описание работы программы……………………………………………4
3. Инструкция пользователя………………………………………………..5
4. Результат работы программы……………………………………………6
5. Блок-схемы программы……………………………….………………..7-9
6. Приложение.………………………………………………….………10-11

**Постановка задачи:**

Разработать программу, реализующую аналог программы «Черепашка» с использованием библиотеки pygame.

* Исполнитель должен находиться на поле некоторого размера и выполнять функции рисования в соответствии с введенными командами;
* Программа должна выполнять следующую последовательность действий:

1. Ввести команду с клавиатуры;

2. Из введенной команды получить ее название и параметры;

3. Выполнить преобразование параметров команды в соотв тип данных;

4. Исполнитель должен выполнить действия соответствующие введенной команде и заданным параметрам;

5. Повторять последовательность выше до ввода команды выхода;

* Команды вводятся с помощью стандартных средств ввода с клавиатуры по одной команде за раз;
* Программа должна реализовывать команды, указанные в таблице ниже;

|  |  |  |
| --- | --- | --- |
| Команда | Описание | Пример |
| up число | Двигаться вверх на заданное количество пикселей | up 10 |
| down число | Двигаться вниз на заданное количество пикселей | down 50 |
| left число | Двигаться влево на заданное количество пикселей | left 10 |
| right число | Двигаться вправо на заданное количество пикселей | right 50 |
| goto x y | Переместиться в указанные координаты | goto 45 68 |
| tail параметр | Поднять/опустить хвост (начать/закончить рисовать)  up – поднять  down - опустить | tail up |

**Описание работы программы:**

Используемые библиотеки: pygame.

Создаём пустой главный экран и в создании рисунка используем команды: up число, down число, left число, right число, goto x y, tail параметр. При вводе команды up координата исполнителя увеличивается по оси у на введенное число. Далее при вводе команды down координата исполнителя уменьшается по оси у на введенное число. Потом при вводе команды left координата исполнителя уменьшается по оси x на введенное число. Следующим действием при вводе команды right координата исполнителя увеличивается по оси x на введенное число. Затем при вводе команды goto координаты исполнителя изменяются по оси x и у на введенное число. По итогу, когда рисунок закончен, закрываем программу командой tail с параметром down.

**Инструкция пользователя:**

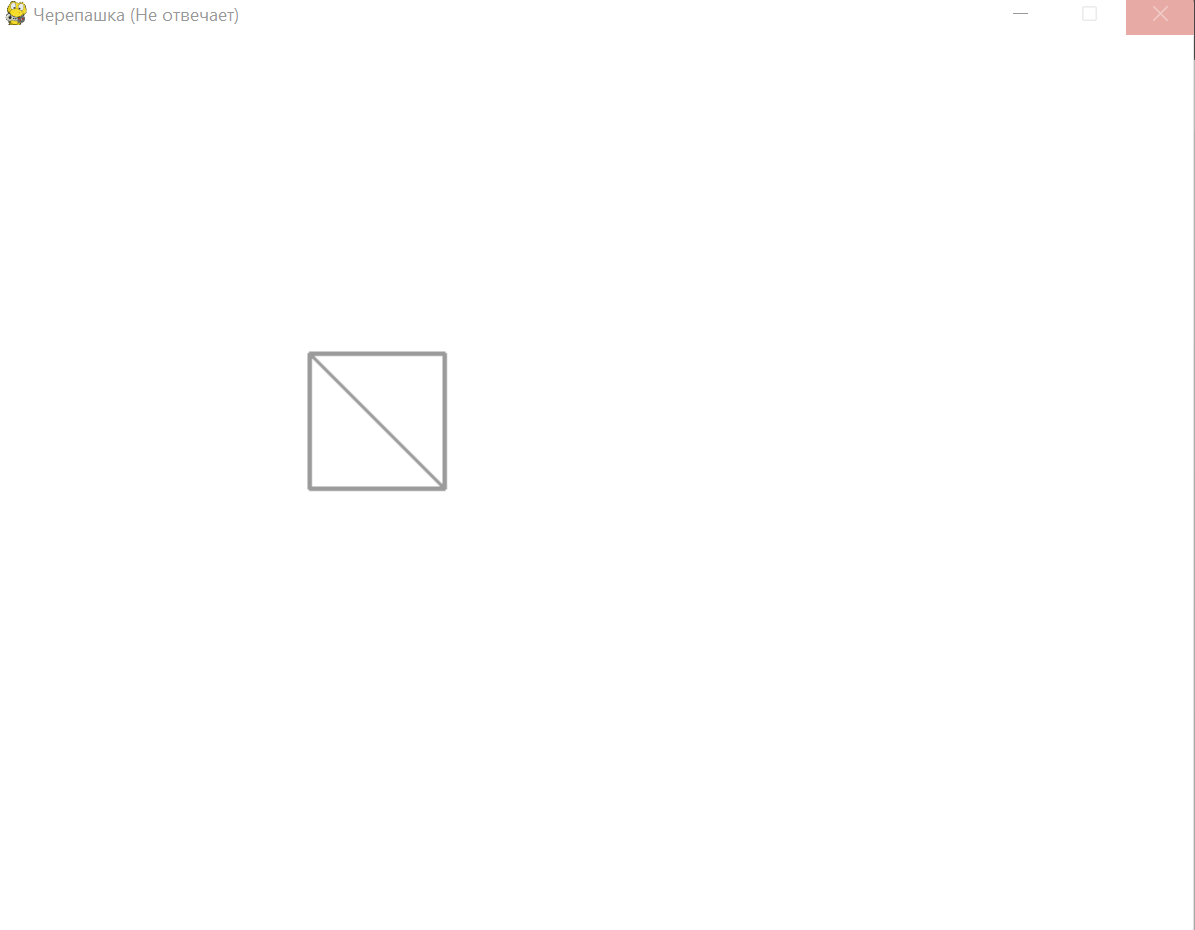
Запуск осуществляется через приложение IDLE. Чтобы начать рисование в диалоговом окне необходимо ввести команду ‘tail up’ и нажать enter. Тем же образом поочередно вписываются остальные команды одной строкой, при этом сама команда и её значение, на которое будет смещаться исполнитель, разделяются пробелом. Исполнитель находится в точке (300; 300). Для завершения процесса рисования вводим команду ‘tail down’.

**Результат работы программы:**

**Изображение выглядит как текст

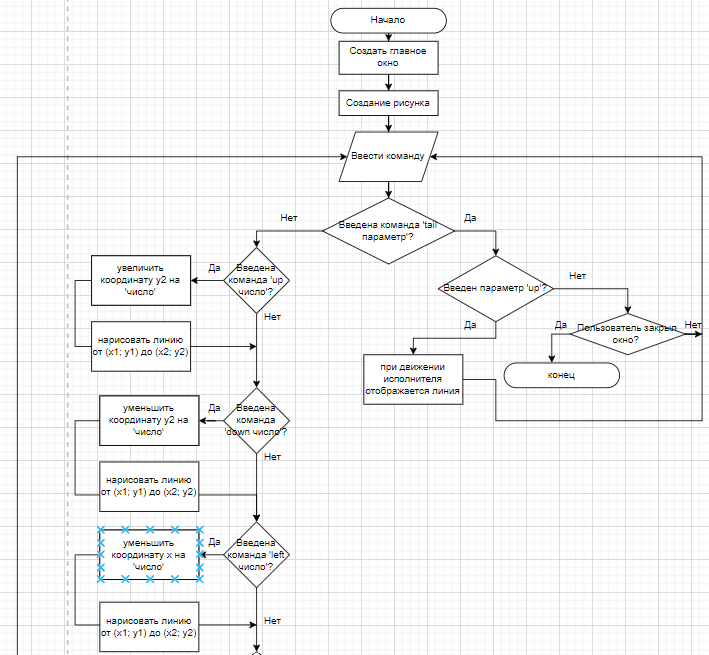
Автоматически созданное описание**

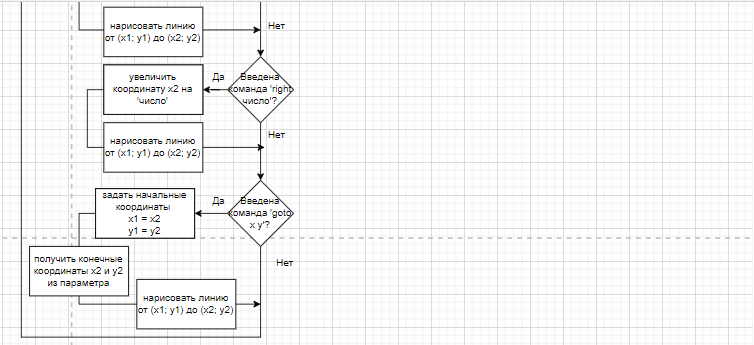
Введенный команды

****

Полученный рисунок

**Блок-схемы программы:**

****

****

**Приложение:**

Листинг файла cherepashka.py

import pygame as pg

pg.init()

WHITE = (255, 255, 255)

BLACK = (0, 0, 0)

COLOR\_LINE = WHITE

screen = pg.display.set\_mode((800, 600))

pg.display.set\_caption("Черепашка")

screen.fill(WHITE)

x1, x2 = 300, 300

y1, y2 = 300, 300

cmds = [x1, y1], [x2, y2]

while True:

for event in pg.event.get():

if event.type == pg.QUIT:

exit()

pg.draw.line(screen, COLOR\_LINE, cmds[0], cmds[1], 3)

pg.display.update()

user = input()

yser = user.split()

user\_up = user.startswith("up")

user\_down = user.startswith("down")

user\_left = user.startswith("left")

user\_right = user.startswith("right")

user\_goto = user.startswith("goto")

user\_tail = user.startswith("tail")

if user\_tail == True:

user\_tail\_up = user.endswith("up")

if user\_tail\_up == True:

COLOR\_LINE = BLACK

elif user\_tail\_up == False:

exit()

else:

if user\_up == True:

x1 = x2

y1 = y2

y2 -= float(yser[-1])

pg.draw.line(screen, COLOR\_LINE, [x1, y1], [x2, y2], 3)

elif user\_down == True:

x1 = x2

y1 = y2

y2 += float(yser[-1])

pg.draw.line(screen, COLOR\_LINE, [x1, y1], [x2, y2], 3)

elif user\_left == True:

x1 = x2

y1 = y2

x2 -= float(yser[-1])

pg.draw.line(screen, COLOR\_LINE, [x1, y1], [x2, y2], 3)

elif user\_right == True:

x1 = x2

y1 = y2

x2 += float(yser[-1])

pg.draw.line(screen, COLOR\_LINE, [x1, y1], [x2, y2], 3)

elif user\_goto == True:

x1 = x2

y1 = y2

x2 = float(yser[-2])

y2 = float(yser[-1])

pg.draw.line(screen, COLOR\_LINE, [x1, y1], [x2, y2], 3)

pg.quit()

quit()