Міністерство освіти та науки України

Національний технічний університет України

«Київський Політехнічний Інститут ім. Ігоря Сікорського»

ФТІ

Кафедра ФТЗЗІ

**Лабораторна робота №9**

з дисципліни: «Програмування 4»

на тему:

**«Побудова об’єктної ієрархії геометричних фігур»**

**Варіант 21**

Виконав:

Ст. гр. ФЕ-81

Пелих Валентин

Перевірив:

доцент Прогонов Д.О.

Київ 2020

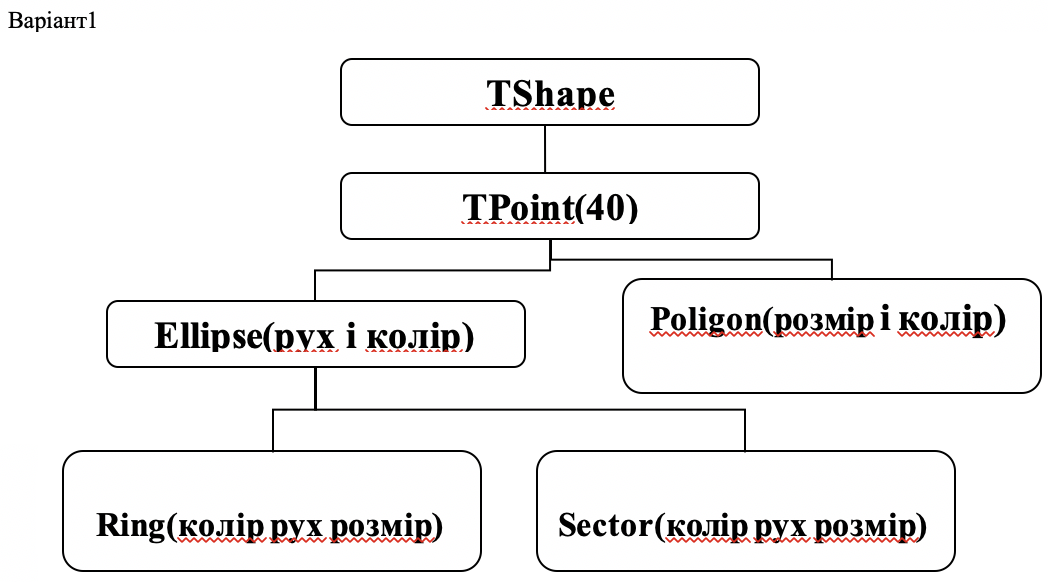
Мета роботи: засвоїти базові поняття ООП на прикладі побудови ієрархії геометричних фігур засобами мови програмування Python.

**1.Виконання лабораторної роботи:**

1.Ознайомтеся з теоретичною частиною;

2.Реалізуйте програмно приклади приведені в тексті;

3.Виконайте свій варіант побудови геометричних фігур за вказівками викладача.



**Код реалізації:**

class TShape (object):

def \_\_init\_\_(self, x, y):

print("TShape-object has created!", "\n")

self.x = float(x)

self.y = float(y)

def display(self):

print("X is:", self.x)

print("Y is:", self.y, '\n')

class TPoint (TShape):

def \_\_init\_\_(self, x, y):

print("TPoint-object has created!")

self.x = float(x)

self.y = float(y)

self.N = 40

print(self.N, "- points has created!", "\n")

class TEllipse (TPoint):

def \_\_init\_\_(self, x, y, a, b, c):

print("TEllipse-object has created!", "\n")

self.xCenter = float(x)

self.yCenter = float(y)

self.major = abs(int(a))

self.minor = abs(int(b))

self.color = str(c)

def display(self):

print("Coordinates of center are:")

print("X is:", self.xCenter)

print("Y is:", self.yCenter)

print("a is:", self.major)

print("b is:", self.minor)

print("Color is:", self.color, '\n')

def changePos(self, newX, newY):

self.xCenter = float(newX)

self.yCenter = float(newY)

print("Coordinates of center have changed!")

print("New X is:", self.xCenter)

print("New Y is:", self.yCenter, "\n")

def changeColor(self, newC):

self.color = str(newC)

print("Color has changed!")

print("New color is:", self.color, "\n")

class TPoligon(TPoint):

def \_\_init\_\_(self, x, y, c, s):

print("TPoligon-object has created!", "\n")

self.xCenter = float(x)

self.yCenter = float(y)

self.color = str(c)

self.size = abs(int(s))

def display(self):

print("Coordinates of center are:")

print("X is:", self.xCenter)

print("Y is:", self.yCenter)

print("Color is:", self.color)

print("Number of angles: ", self.size, '\n')

def changeSize(self, newS):

self.size = abs(int(newS))

print("Size has changed!")

print("New size is:", self.size, "\n")

def changeColor(self, newC):

self.color = str(newC)

print("Color has changed!")

print("New color is:", self.color, "\n")

class TRing(TEllipse):

def \_\_init\_\_(self, x, y, r, t, c):

print("TRing-object has created!", "\n")

self.xCenter = float(x)

self.yCenter = float(y)

self.color = str(c)

self.intR = abs(int(r))

self.thickness = abs(int(t))

def display(self):

print("Coordinates of center are:")

print("X is: ", self.xCenter)

print("Y is: ", self.yCenter)

print("Color is: ", self.color)

print("Radius is: ", self.intR)

print("Thickness is: ", self.thickness, '\n')

# changePos() for TRing has already exist,

# because there is inheritance: TEllipse -> TRing

# changeColor() for TRing has already exist,

# because there is inheritance: TEllipse -> TRing

def changeRadius(self, newR):

self.intR = abs(int(newR))

print("Radius has changed!")

print("New radius is: ", self.intR, "\n")

class TSector(TEllipse):

def \_\_init\_\_(self, x, y, l, a, c):

print("TRing-object has created!", "\n")

self.xCenter = float(x)

self.yCenter = float(y)

self.color = str(c)

self.alpha = float(a)

self.length = abs(int(l))

def display(self):

print("Coordinates of center are:")

print("X is: ", self.xCenter)

print("Y is: ", self.yCenter)

print("Color is: ", self.color)

print("Angle is: " , self.alpha)

print("Length is: ", self.length, '\n')

# changePos() for TRing has already exist,

# because there is inheritance: TEllipse -> TRing

# changeColor() for TRing has already exist,

# because there is inheritance: TEllipse -> TRing

def changeLength(self, newL):

self.length = abs(int(newL))

print("Length has changed!")

print("New length is: ", self.length, "\n")

def changeAlpha(self, newA):

self.alpha = abs(int(newA))

print("Length has changed!")

print("New abgle is: ", self.alpha, "\n")

if \_\_name\_\_ == "\_\_main\_\_":

S = TShape(10, 5)

S.display()

P = TPoint(-10, 5)

P.display()

E = TEllipse(11, 11, 2, 3, "Blue")

E.display()

E.changeColor("Black")

E.changePos(-11, -11)

E.display()

Po = TPoligon(2, 4, "Red", 15)

Po.display()

Po.changeColor("Green")

Po.changeSize(25)

Po.display()

R = TRing(35, 67, 6, 2, "Yellow")

R.display()

R.changePos(12, 12)

R.changeColor("Cyan")

R.changeRadius(12)

R.display()

S = TSector(7, 5, 20, 90, "Purple")

S.display()

S.changeColor("Pink")

S.changeLength(30)

S.changeAlpha(180)

S.changePos(9, 9)

S.display()