КИЇВСЬКИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

**Лабораторна робота №3**

З теорії ймовірності та математичної статистики

Виконав:

студент ІІ курсу ФІОТ

група ІО-32

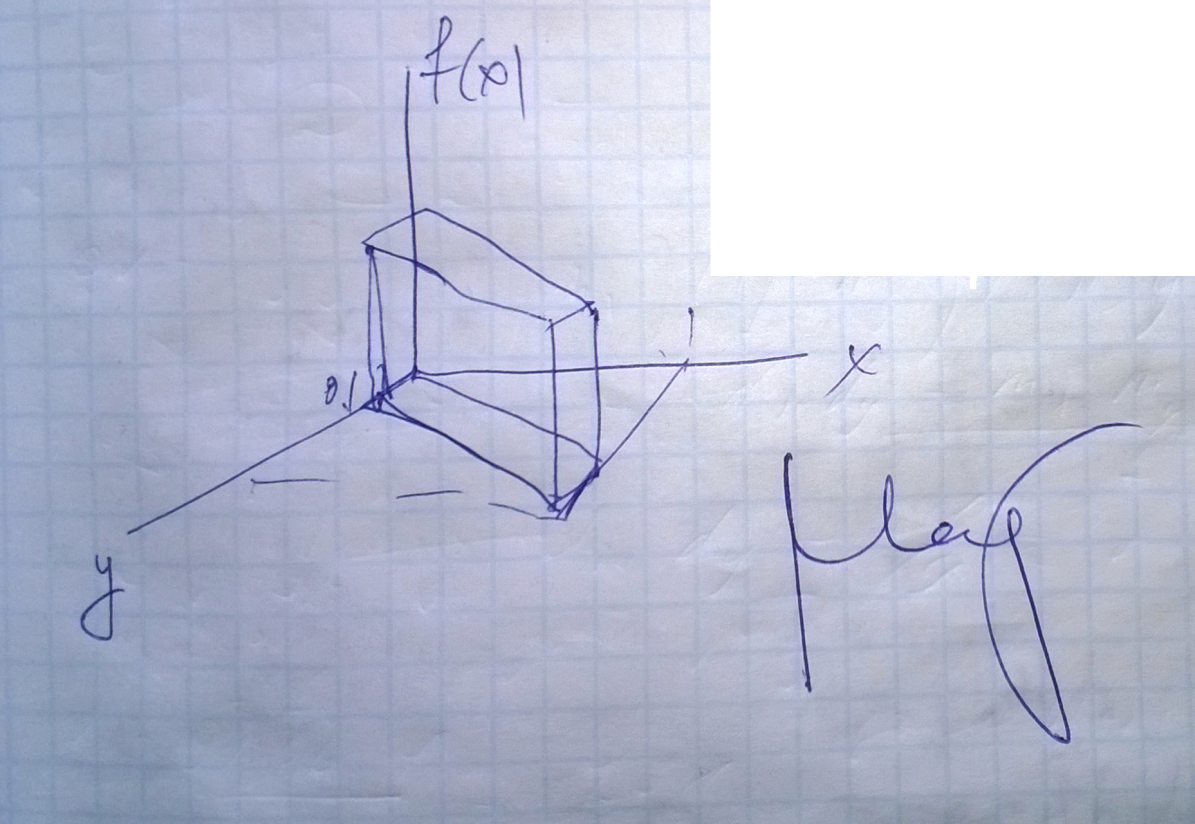
Попенко Руслан

Перевірив:

Марковський О. П.

Київ-2014

**Завдання**

****

**Лістинг програми**

**package** laba3;

**public** **class** **Lab3** {

**public** **static** **void** **main**(**String**[] args) {

**NewFigure** **MyFig** = **new** NewFigure(1, 1);

MyFig.Gen(5000);

**System**.***out***.println("Mx= " + MyFig.MatOchX());

**System**.***out***.println("My= " + MyFig.MatOchY());

**System**.***out***.println("Dx= " + MyFig.DispersionX());

**System**.***out***.println("Dy= " + MyFig.DispersionY());

**System**.***out***.println("Sigma x= "+ MyFig.SeredneKvadrX());

**System**.***out***.println("Sigma y= "+ MyFig.SeredneKvadrY());

**System**.***out***.println("Cov= " + MyFig.Covar());

**System**.***out***.println("Ro= " + MyFig.Corel());

}

}

**package** laba3;

**import** java.util.Random;

**public** **class** **Form** {

// bottom of the shape

**private** **double** a;

**private** **double** b;

**public** **static** **double** *epsilon* = 0.0001;

**public** **Form**(**double** a, **double** b) {

**this**.a = a;

**this**.b = b;

}

**public** **Massiv** **generateVector**() {

**double** **x** = generateX();

**double** **y** = generateY(x);

**return** **new** Massiv(x, y);

}

**private** **double** **generateX**() {

**Random** **rand** = **new** Random();

**double** **r** = rand.nextDouble();

**double** **square** = 0;

**double** **step** = 0.000001;

**double** **xResult** = 0.0;

**double** **trapeciaSquare** = 0;

**while** (square < r - *epsilon*) {

trapeciaSquare = ((functionPhiX(xResult) + functionPhiX(xResult

+ step)) / 2)

\* step;

square += trapeciaSquare;

xResult += step;

}

**return** xResult;

}

**private** **double** **generateY**(**double** x) {

**Random** **rand** = **new** Random();

**double** **r** = rand.nextDouble();

**double** **square** = 0;

**double** **step** = 0.000001;

**double** **yResult** = 0.0;

**double** **trapeciaSquare** = 0;

**while** (square < r) {

trapeciaSquare = (functionF\_if\_X(yResult, x) + functionF\_if\_X(

yResult + step, x)) / 2 \* step;

square += trapeciaSquare;

yResult += step;

}

**return** yResult;

}

**private** **double** **functionPhiX**(**double** x) {

**return** 1;

}

**private** **double** **functionF\_if\_X**(**double** y, **double** x) {

**return** 10;

}

}

**package** laba3;

**import** java.util.ArrayList;

**public** **class** **NewFigure** {

**private** **Form** shape;

**private** **ArrayList**<Massiv> numbers;

**private** **double** mathX;

**private** **double** mathY;

**private** **double** dispersionX;

**private** **double** dispersionY;

**private** **double** cov;

**public** **NewFigure**(**double** a, **double** b) {

shape = **new** Form(a, b);

numbers = **new** ArrayList<Massiv>();

}

**public** **void** **Gen**(**int** count) {

**long** **startTime** = **System**.*nanoTime*();

**long** **currentTime** = 0;

**long** **totalTime** = 0;

**long** **totalSecond**;

**int** **n** = 0;

**for** (**int** **i** = 0; i < count; i++) {

numbers.add(shape.generateVector());

currentTime = **System**.*nanoTime*();

totalTime = currentTime - startTime;

totalSecond = totalTime / 1000000000l;

**if** (totalSecond > n) {

n++;

}

}

}

**public** **double** **MatOchX**() {

mathX = 0;

**for** (**Massiv** **v** : numbers) {

mathX += v.getX();

}

mathX /= numbers.size();

**return** mathX;

}

**public** **double** **MatOchY**() {

mathY = 0;

**for** (**Massiv** **v** : numbers) {

mathY += v.getY();

}

mathY /= numbers.size();

**return** mathY;

}

**public** **double** **DispersionX**() {

dispersionX = 0;

**for** (**Massiv** **v** : numbers) {

dispersionX += (v.getX() - mathX) \* (v.getX() - mathX);

}

dispersionX /= numbers.size();

**return** dispersionX;

}

**public** **double** **DispersionY**() {

dispersionY = 0;

**for** (**Massiv** **v** : numbers) {

dispersionY += (v.getY() - mathY) \* (v.getY() - mathY);

}

dispersionY /= numbers.size();

**return** dispersionY;

}

**public** **double** **SeredneKvadrX**() {

**return** **Math**.*sqrt*(dispersionX);

}

**public** **double** **SeredneKvadrY**() {

**return** **Math**.*sqrt*(dispersionY);

}

**public** **double** **Covar**() {

cov = 0;

**for** (**Massiv** **v** : numbers) {

cov += (v.getX() - mathX) \* (v.getY() - mathY);

}

cov /= numbers.size();

**return** cov;

}

**public** **double** **Corel**() {

**return** cov / (**Math**.*sqrt*(dispersionX) \* **Math**.*sqrt*(dispersionY));

}

}

**package** laba3;

**public** **class** **Massiv** {

**private** **double** x;

**private** **double** y;

**public** **double** **getX**() {

**return** x;

}

**public** **void** **setX**(**double** x) {

**this**.x = x;

}

**public** **double** **getY**() {

**return** y;

}

**public** **void** **setY**(**double** y) {

**this**.y = y;

}

**public** **Massiv**(**double** x, **double** y) {

**super**();

**this**.x = x;

**this**.y = y;

}

}