*Національний технічний університет України*

*‘Київський політехнічний інститут’*

*Факультет інформатики та обчислювальної техніки*

*Кафедра обчислювальної техніки*

***Лабораторна робота №4***

*“ Розв’язання нелінійних рівнянь на ЕОМ ”*

*Виконав: Мроць Ю.Б.*

*Група: ІО-12*

*Номер ЗК: 1219*

*Київ — 2013*

***Мета***

Метою даного заняття є ознайомлення з методиками та вивчення різних алгоритмів розв’язання нелінійних рівнянь на ЕОМ.

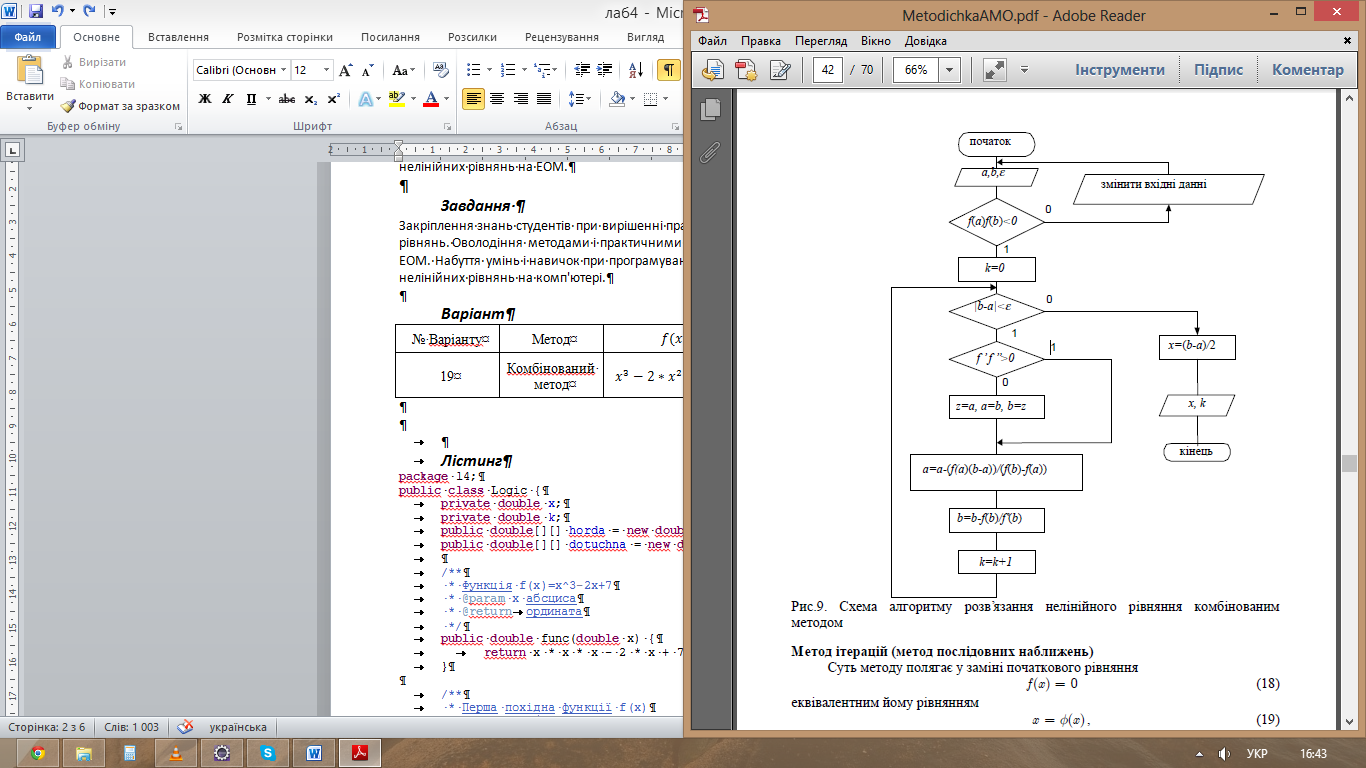
***Завдання***

Закріплення знань студентів при вирішенні практичних завдань з розв’язування нелінійних рівнянь. Оволодіння методами і практичними навичками розв’язування нелінійних рівнянь на ЕОМ. Набуття умінь і навичок при програмуванні та налагодженні програм для розв’язування нелінійних рівнянь на комп'ютері.

***Варіант***

|  |  |  |  |
| --- | --- | --- | --- |
| № Варіанту | Метод |  | Примітка |
| 19 | Комбінований метод |  |  |

***Блок-схема алгоритму пошуку кореня комбінованим методом***



***Лістинг***

**package** l4;

**public** **class** Logic {

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

**private** **double** k;

**public** **double**[][] horda = **new** **double**[100][100];

**public** **double**[][] dotuchna = **new** **double**[100][100];

/\*\*

\* Функція f(x)=x^3-2x+7

\* **@param** x абсциса

\* **@return** ордината

\*/

**public** **double** func(**double** x) {

**return** x \* x \* x - 2 \* x + 7;

}

/\*\*

\* Перша похідна функції f(x)

\* **@param** x абсциса

\* **@return** ордината

\*/

**public** **double** d\_func(**double** x) {

**return** 3 \* x \* x - 2;

}

/\*\*

\* Друга похідна функції f(x)

\* **@param** x абсциса

\* **@return** ордината

\*/

**public** **double** d2\_func(**double** x) {

**return** 6 \* x;

}

**public** **void** find\_x(**double** a, **double** b, **double** e) {

setK(0);

**if** (func(a) \* func(b) >= 0) {

a -= 0.1;

b += 0.1;

}

**int** i = 0;

**while** (Math.*abs*(b - a) >= e) {

horda[0][i] = a; // xa

horda[0][i+1] = func(a); // ya

horda[1][i] = b; // xb

horda[1][i+1] = func(b); // yb

**if** (d\_func((a + b) / 2) \* d2\_func((a + b) / 2) <= 0) {

**double** z = a;

a = b;

b = z;

}

a -= (func(a) \* (b - a)) / (func(b) - func(a));

dotuchna[0][i] = b; // xa

dotuchna[0][i+1] = func(b); // ya

b -= func(b) / d\_func(b);

dotuchna[1][i] = b; // xb

dotuchna[1][i+1] = 0; // yb

k++;

i++;

i++;

}

x = (a + b) / 2;

}

**public** **double** getK() {

**return** k;

}

**public** **void** setK(**double** k) {

**this**.k = k;

}

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

**return** x;

}

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

**this**.x = x;

}

}

**package** l4;

**public** **class** Test {

**public** **static** **double** *a*; // ліва межа відрізка, на якому знаходиться корінь

**public** **static** **double** *b*; // права межа відрізка, на якому знаходиться корінь

**public** **static** **double** *e*; // точність обчислення кореня

**public** **static** StartWindow *st*;

**public** **static** Logic *l*;

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

*st* = **new** StartWindow();

*l* = **new** Logic();

**while** (!*st*.cont) {

System.*out*.print(" ");

}

*a* = Double.*parseDouble*(*st*.getA());

*b* = Double.*parseDouble*(*st*.getB());

*e* = Double.*parseDouble*(*st*.getE());

*l*.find\_x(*a*, *b*, *e*);

XYSeries series1 = **new** XYSeries("y = x^3-2\*x^2+7");

**for** (**double** j = -10; j <= 10; j += 0.01){

series1.add(j, *l*.func(j));

}

XYSeries series2 = **new** XYSeries("y = x^3-2\*x^2+7");

**for** (**double** j = *a*; j <= *b*; j += 0.01){

series2.add(j, *l*.func(j));

}

XYSeriesCollection data = **new** XYSeriesCollection(series1);

XYSeriesCollection data2 = **new** XYSeriesCollection(series2);

**for** (**int** i = 0; i < *l*.getK() \* 2; i += 2) {

XYSeries h = **new** XYSeries("Хорда " + (i / 2 + 1));

XYSeries d = **new** XYSeries("Дотична " + (i / 2 + 1));

h.add(*l*.horda[0][i], *l*.horda[0][i+1]);

h.add(*l*.horda[1][i], *l*.horda[1][i+1]);

d.add(*l*.dotuchna[0][i], *l*.dotuchna[0][i + 1]);

d.add(*l*.dotuchna[1][i], *l*.dotuchna[1][i + 1]);

data2.addSeries(h);

data2.addSeries(d);

}

JFreeChart chart1 = ChartFactory.*createXYLineChart*(

"Функція f(x)=x^3-2\*x^2+7",

"x", "y",

data,

PlotOrientation.*VERTICAL*,

**true**, **true**, **true**);

JFreeChart chart2 = ChartFactory.*createXYLineChart*(

"Візуалізація покрокового знаходження кореня",

"x", "y",

data2,

PlotOrientation.*VERTICAL*,

**true**, **true**, **true**);

ChartPanel chartPanel1 = **new** ChartPanel(chart1);

ChartPanel chartPanel2 = **new** ChartPanel(chart2);

JFrame f = **new** JFrame();

f.setLayout(**new** BorderLayout());

f.add(chartPanel1, BorderLayout.*WEST*);

f.add(chartPanel2, BorderLayout.*EAST*);

JPanel panel = **new** JPanel();

panel.setBorder(**new** LineBorder(**new** Color(0, 0, 0)));

f.getContentPane().add(panel);

panel.setLayout(**new** GridLayout(2, 0));

panel.add(**new** JLabel("x = " + (Math.*round*(*l*.getX() / *e*) \* *e*), JLabel.*CENTER*));

panel.add(**new** JLabel("k = " + *l*.getK(), JLabel.*CENTER*));

f.add(panel, BorderLayout.*SOUTH*);

f.pack();

f.setVisible(**true**);

f.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);

}

}

**package** l4;

**public** **class** StartWindow **extends** JFrame {

**private** **static** **final** **long** *serialVersionUID* = 1L;

**private** JTextField textField\_a;

**private** JTextField textField\_b;

**private** JTextField textField\_e;

**private** String a = "-3";

**private** String b = "-2";

**private** String eps = "0.0001";

**public** **boolean** cont = **false**;

**private** Logic l = **new** Logic();

/\*\*

\* Create the frame.

\*/

**public** StartWindow() {

setTitle("");

setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);

setBounds(100, 100, 600, 290);

getContentPane().setLayout(**new** GridLayout(3, 1, 0, 0));

JLabel lbla = **new** JLabel("");

lbla.setBorder(**new** MatteBorder(1, 1, 0, 1, (Color) **new** Color(0, 0, 0)));

lbla.setHorizontalTextPosition(SwingConstants.*CENTER*);

lbla.setHorizontalAlignment(SwingConstants.*CENTER*);

getContentPane().add(lbla);

JPanel panel = **new** JPanel();

panel.setBorder(**new** LineBorder(**new** Color(0, 0, 0)));

getContentPane().add(panel);

panel.setLayout(**new** GridLayout(0, 2, 0, 0));

JLabel label\_a = **new** JLabel("");

label\_a.setBorder(**new** MatteBorder(0, 0, 1, 1, (Color) **new** Color(0, 0, 0)));

label\_a.setHorizontalAlignment(SwingConstants.*CENTER*);

panel.add(label\_a);

textField\_a = **new** JTextField();

textField\_a.setText("-3");

textField\_a.addKeyListener(**new** KeyAdapter() {

@Override

**public** **void** keyReleased(KeyEvent arg0) {

setA(textField\_a.getText());

}

});

textField\_a.setBorder(**new** MatteBorder(0, 0, 1, 0, (Color) **new** Color(0, 0, 0)));

textField\_a.setToolTipText("");

textField\_a.setHorizontalAlignment(SwingConstants.*CENTER*);

panel.add(textField\_a);

textField\_a.setColumns(10);

JLabel label\_b = **new** JLabel("");

label\_b.setBorder(**new** MatteBorder(0, 0, 1, 1, (Color) **new** Color(0, 0, 0)));

label\_b.setHorizontalAlignment(SwingConstants.*CENTER*);

panel.add(label\_b);

textField\_b = **new** JTextField();

textField\_b.setText("-2");

textField\_b.addKeyListener(**new** KeyAdapter() {

@Override

**public** **void** keyReleased(KeyEvent e) {

setB(textField\_b.getText());

}

});

textField\_b.setBorder(**new** MatteBorder(0, 0, 1, 0, (Color) **new** Color(0, 0, 0)));

textField\_b.setHorizontalAlignment(SwingConstants.*CENTER*);

panel.add(textField\_b);

textField\_b.setColumns(10);

JLabel label\_e = **new** JLabel("\u0422\u043E\u0447\u043D\u0456\u0441\u0442\u044C (e)");

label\_e.setBorder(**new** MatteBorder(0, 0, 0, 1, (Color) **new** Color(0, 0, 0)));

label\_e.setHorizontalAlignment(SwingConstants.*CENTER*);

panel.add(label\_e);

textField\_e = **new** JTextField();

textField\_e.setText("0.0001");

textField\_e.addKeyListener(**new** KeyAdapter() {

@Override

**public** **void** keyReleased(KeyEvent e) {

setE(textField\_e.getText());

}

});

textField\_e.setBorder(**new** MatteBorder(0, 0, 0, 0, (Color) **new** Color(0, 0, 0)));

textField\_e.setHorizontalAlignment(SwingConstants.*CENTER*);

panel.add(textField\_e);

textField\_e.setColumns(10);

JPanel panel\_OK = **new** JPanel();

panel\_OK.setName("");

panel\_OK.setToolTipText("");

getContentPane().add(panel\_OK);

panel\_OK.setLayout(**null**);

JButton button\_OK = **new** JButton("\u041E\u041A");

button\_OK.addMouseListener(**new** MouseAdapter() {

@Override

**public** **void** mouseReleased(MouseEvent arg0) {

**if** (l.func(Double.*parseDouble*(getA())) \* l.func(Double.*parseDouble*(getB())) < 0) {

cont = **true**;

} **else** {

JOptionPane.*showMessageDialog*(**null**,

"На відрізку [" + getA() + "; " + getB() + "] немає кореня.");

}

}

});

button\_OK.setBorder(**new** LineBorder(**new** Color(0, 0, 0)));

button\_OK.setHorizontalTextPosition(SwingConstants.*CENTER*);

button\_OK.setBounds(258, 26, 80, 25);

panel\_OK.add(button\_OK);

setVisible(**true**);

}

**public** String getA() {

**return** a;

}

**public** **void** setA(String a) {

**this**.a = a;

}

**public** String getB() {

**return** b;

}

**public** **void** setB(String b) {

**this**.b = b;

}

**public** String getE() {

**return** eps;

}

**public** **void** setE(String e) {

**this**.eps = e;

}

}