МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
„КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

Базові алгоритми числового аналізу

Методичні вказівки до виконання лабораторних робіт з
дисципліни „Числові методи”
для студентів спеціальності „Автоматизоване управління
технологічними процесами”
напряму „Автоматизація та комп'ютерно-інтегровані технології”

Рекомендовано Вченою радою інженерно-хімічного факультету

Київ
НТУУ «КПІ»
2014

Базові алгоритми числового аналізу: Метод. вказівки до виконання лабораторних робіт з дисципліни „Числові методи” для студентів спеціальності „Автоматизоване управління технологічними процесами” напряму „Автоматизація та комп'ютерно-інтегровані технологічні комплекси” / Уклад.: О.В. Ситніков”, 2014. – 42с.

Гриф надано Вченою радою ІХФ

(Протокол № від 2014р.)

Навчальне видання

БАЗОВІ АЛГОРИТМИ ЧИСЛОВОГО АНАЛІЗУ
Методичні вказівки до виконання лабораторних робіт з дисципліни „Числові методи” для студентів спеціальності „Автоматизоване управління технологічними процесами” напряму „Автоматизація та комп'ютерно-інтегровані технології”

Укладачі: Ситніков Олексій Володимирович

Відповідальний редактор А.І.Жученко, д-р техн.наук, проф.

Рецензент : В.І. Сівецький, к.т.н., проф.

Авторська редакція

Зміст

Вступ

В даному методичному посібнику розглянуто основи роботи з алгоритмічною мовою програмування *Pascal*. Це дає можливість на практиці вирішити поставлені задачі. Робота відбувається в текстовому та графічному режимах, за допомогою підключення відповідних модулів.

Перші 4 роботи присвячені роботі з текстовим режимом (оператори, підпрограми). Розглянуто розрахунок значень функції в циклі, заповнення масиву, робота із заповненим масивом.

Наступні роботи призначені для ознайомлення студентів з графічним режимом *Pascal*. В п’ятій роботі необхідно побудувати графік функції однієї змінної. Наступна робота дає зрозуміти принцип побудови функції заданої параметрично – так званий годограф.

В роботі сім будується система ізоліній, тобто система ліній однакових значень функції двох змінних та виводу на екран графічного результату.

Всі лабораторні роботи доповнені текстами демонстраційно-відлагоджувальних програм, тому даний посібник може бути використаний для самостійної роботи, але слід звернути уваги на те, що при виникненні запитань під час самостійної роботи не буде можливості їх задавати. Всі програми робочі і автори звертають увагу на можливість студентів самостійно вносити зміни до програм, вдосконалювати їх.

Лабораторна робота №1

Введення в *Pascal*.

Мета роботи : Дослідити середовище програмування *Pascal*, призначення основних команд меню.

Теоретичні відомості.

Мова *Pascal* пристосована для використання на сучасних персональних комп'ютерах типа *IBM PC*.

Команди основного меню `Pascal` (в дужках приведені "гарячі клавіші" виконання команд):

Основное меню складається з переліка команд, які виконують компоненти систем програмування: *File*, *Edit*, *Run*, *Compile*, *Options*, *Debug*, *Break*/*watch*. Кожна з них, за виключенням Edit, має додаткове меню, що зя'вляється на екрані після звертання до основної команди.

Команда *File* завантажує вміст файла, зберігає файл, звертається до каталогів, виходить в операційну систему.

Підменю містить 9 команд: *Load*, *Pick*, *New*, *Save*, *Write to*, *Directory*, *Change dir*, *OS shell*, *Quit*.

-*Load*- команда завантаження файла в оперативну пам'ять та вікно редактування (<*F*3>) .

- *Pick*- команда завантаження файла, що знаходиться в списку останніх восьми файлів, що завантажувались у вікно редагування файла на протязі поточного сеанса роботи в *Pascal*.

- *New*- команда створення нового файла. В результаті її виконання очищується оперативна пам'ять, вікно редагування, встановлюється ім'я файла *NONAME.PAS* в інформаційному рядку екрана.

- *Save*- команда збереження файла (<*F*2>) .

- *Write to*- команда збереження файла під вказанним ім'ям.

- *Directory*- команда виводу вмісту вказаного каталогу.

- *Change Dir*- зміна поточного каталогу.

- *Quit*- команда завершує роботу в *Pascal* та передає керування операційній системі (<*Alt-X*>) .

Команда *Edit* переводить ТП в режим редагування файла.

Команда *Run* дозволяє виконувати та трасувати програму. Підменю містить 6 команд (розглянемо необхідні в данній роботі):

- *Run*- команда виконання попередньо відкомпільованої програми. Якщо перед виконанням в тексті програми вносилися зміни, перед виконанням буде автоматично виконана компіляція (<*Ctrl-F*9>) .

- *User Screen*- перехід в екран користувача. Команда дозволяє побачити на екрані результат виконання програми (<*Alt-F*5>).

- *Program Reset*- команда відміняє поточний сеанс відладки, звільнює пам'ять, зайняту програмою, закриває всі відкриті програмою файли, прибирає поточну границю виконання (<*Ctrl-F*2>) .

Команда Compile -компіляція самостійної програмної одиниці (програми чи модуля) (<*Alt-F*9>).

Команда Break/watch призначена для установки та відміни точок зупинки, для додавання, редагування та видаления записів з вікна *Watch*.

Команда *Debug* призначена для відладки.

Деякі клавіші, які необхідні в роботі:

<F1> - (Help) Отримання контекстно-залежної довідки (допомоги);

<F5> - (Window/Zoom) Збільшення/зменшення розмірів вікна;

<Alt-F> - Активизація головного меню.

Програма на *Pascal* складається із заголовка, описової та операторної частини. Заголовок програми починається словом program.

В описовій частині відбувається об'явлення типів (*type*), змінних (*var*), констант (*const*), міток (*label*), підпрограм (*procedure, function*).

Операторна частина починається словом *begin*, а закінчується *end.*

А теперь еще несколько слов о процедурах *write* и *read* (все нижесказанное полностью относится и к их модификациям *writeln* и *readln*).

Write() може виводити текстову інформацію (в лапках ‘…’) та числову значення змінних : write(A) або write(‘A=’,A).

Writeln(…) – перевід курсору на новий рядок після виконання процесу виводу.

read() – зчитати в значення змінної в дужках те, що введено з клавіатури.

Одним з операторів, є оператор умовного переходу:

if < логічний вираз > then < оператор >.

Спочатку перевіряється < логічний вираз > (умова). Якщо логічний вираз - *true*, то виконується < оператор >, якщо логічний вираз *false*, то виконується наступний оператор.

Повний оператор.

if < логічний вираз > then < оператор 1 > else < оператор 2 >.

Спочатку перевіряється < логічний вираз > (умова). Якщо логічний вираз - *true*, то виконується < оператор 1 >, якщо логічний вираз *false*, то виконується < оператор 2 >.

Хід роботи.

1. Завантажити середовище *Pascal* (з віконки на робочому столі).
2. Створити новий файл (меню *File* → команда *New*).
3. Зберегти файл під назвою *LR*11(меню *File* → команда *Save*)
4. Набрати текст програми :

```
Program LR11;
    Var  b : integer;
         a,x : real;
    Begin
       b:=20;
       a:=2.47
       x:=sqr(b)*a;
       writeln(‘x=’,x)
     End.
```

5. Відкомпілювати програму (<*Ctrl-F9*>).

6. Зберегти файл з новою назвою *LR*12 (меню *File* → команда *Write to*).

```
Program LR12;
        Var a : integer;
             y : real;
        Begin
            writeln('введите a=');
            read(a);
            if  a>0 then y:=sqrt(a);
            writeln('y=',y)
         End.
```

7. Відкомпілювати програму (<*Ctrl-F*9>).

8. y=_____.

9. Написати програму по розрахунку наступного виразу : $y=tg(a^2+\sqrt{b})-e^{|cos(c)|}$, при умові, що *a*, *b*, *c* константи і дорівнюють відповідно 1.2, 3.7, 4.7. Вивести результат на екран.

y=_____.

10. Написати програму по розрахунку $y=log_2(x)$, якщо $x>0{,}5$ та $y=sin(x)$ в інших випадках, *x* вводити з клавіатури.

Вивести результат на екран. Занести результат у таблицю.

X_____. Y_____.

Звіт повинен містити назву роботи, мету, короткі теоретичні відомості, значення функції *y* в п.8,9; програми з п.9 та 10.

Контрольні запитання

1. В чому відмінність *Write*() від *Writeln*() ?
2. Пояснити принцип роботи оператора *if*?
3. Які є стандарті функції *Pascal*?
4. Як зміниться вивід змінних на екран при заміні *writeln*('*x*=',*x*) на *writeln*(*x*)

Лабораторна робота №2

Оператори циклу

Мета роботи : Дослідити операції, які виконуються з операторами циклу. Навчитись працювати з операторами циклу.

Теоретичні відомості.

В *Pascal* передбачено 3 цикли: арифметичний, з після умовою, з перед умовою.

Арифметичний цикл

for < параметр(змінна) циклу >: = < початкове значення > to < кінцеве значення > do < оператор >;

у випадку коли < початкове значення > менше < кінцеве значення >

for < параметр(змінна) циклу >: = < початкове значение > downto < кінцеве значення > do < оператор >;

у випадку коли < початкове значення > більше < кінцеве значення >

< параметр(змінна) циклу >, < початкове значення > , < кінцеве значення > - тільки цілого типу

Цикл з перед умовою

while < умова зупинки циклу> do < оператор >.

Цикл з після умовою

repeat
< тіло циклу >
until < умова виходу >.

Хід роботи.

1. Завантажити середовище *Pascal* (з віконки на робочому столі).
2. Створити новий файл (меню *File* → команда *New*).
3. Зберегти файл під назвою *LR*11(меню *File* → команда *Save*)
4. Набрати текст програми :

```
Program LR11;
Var a, b : integer;
      x : real;
   Begin
```

```
b:=20;
for a:=1 to 10 do
 begin
    x:=sqr(b)+sin(sqrt(a));
    writeln('x=',x)
End.
```

5. Відкомпілювати програму (<*Ctrl-F9*>).

6. Занести значення *x* до протоколу

_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

7. Внести зміни до програми використовуючи цикл з перед умовою.

Початкове значення *a*=1, кінцеве значення *a*=10, крок зміни *a*=0.5

8. Занести результати роботи програми до протоколу

Значення *x :*

_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

9. Внести зміни до програми використовуючи цикл з після умовою.

Початкове значення *a*=1, кінцеве значення *a*=10, крок зміни *a*=0.5

10. Занести результати роботи програми до протоколу

Значення *x :*

_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

Звіт повинен містити назву роботи, мету, короткі теоретичні відомості, значення *x* в п.6, 8, 10; програми з п.7 та 9

Контрольні запитання

1. Які є стандарті цикли *Pasacl* ?
2. Пояснити принцип роботи арифметичного циклу?
3. Пояснити принцип роботи циклу з після умовою?
4. Пояснити принцип роботи циклу з перед умовою?

Лабораторна робота №3

Робота з масивами.

Мета роботи : Дослідити операції, які виконуються з масивами.

Теоретичні відомості.

Масиви (матриці) можуть бути одно- або багатовимірні, тобто бути розмірністю 1 рядок на декілька стовпців (1 стовпець на декілька рядків) або декілька стовпців на декілька рядків.

В переліку змінних записуються :

одномірні масиви – *A*:*array*[-1..31] *of real*;

багатомірні масиви *B*:*array*[1..10, 1.. 15] *of real*,

де *A*,*B* – змінна типу масив, *array* – ключове слово, яке означає масив, [1..10] – вказує на кількість елементів масиву, *real* – тип змінних, що записуються у масив.

В загальному випадку масив являє собою таблицю для зберігання значень змінних, що логічно віднесені до якоїсь окремої групи і потрібен для швидкого доступу до цих даних.

В масив можна заносити значення кількості елементів. Прийнято для зручності, що кількість елементів масиву заноситься у комірку з індексом -1. В нульову комірку заноситься перший елемент послідовності. В розглядуваних задачах в масив будуть заноситися коефіцієнти полінома, тобто в нульову комірку заноситься коефіцієнт, що стоїть перед змінною в степені 0, в першу – в степені 1 і так далі до коефіцієнта, що стоїть перед змінною в степені *n*.

Хід роботи.

1. Завантажити середовище *Pascal*.
2. Набрати текст програми по вводу елементів масиву з клавіатури та виводу елементів масиву на екран:

```
Program MAS1;
  Uses Crt;
  Var s,n:integer;
```

```
        A:  array [-1..30] of real;
    begin
      Write('Введите количество элементов массива (от 15 до 30) – n=')
      Read(n);
       A[-1]:=n;
       For s:=0 to n do
         begin
           Write('Элемент   ',s,'=');
            Read(A[s])
         end;
        for s:=0 to n do
            Writeln('A[',s,']=',A[s]:4:2)
    end.
```

3. Відкомпілювати програму.
4. Переглянути та перенести результат в протокол.

n= .

s																											
A[s]																											

5. Написати програму по розрахунку суми елементів масиву. Вивести результат на екран. Перенести результат в протокол.

Сума елементів масиву=____.

6. Написати програму по розрахунку кількості негативних елементів в масиві. Вивести результат на екран. Перенести результат в протокол.

Кількість негативних елементів=____.

Звіт повинен містити назву роботи, мету, короткі теоретичні відомості, тексти виконуваних програм п.2, 5, 6 та їх результати роботи.

Контрольні запитання

1. В чому особливість роботи з масивом *A* коли *A*: *array*[-1..30] *of char*;

2. Як зміниться вивід елементів масиву на екран при заміні *writeln*(‘*A*[‘,*s*,’]=’,*A*[*s*]:4:2) на *writeln*(‘*A*[*s*]=’,*A*[*s*]:4:2) ?

3. При заміні *A*: *array*[-1..30] *of real* на *A*: *array*[3..30] *of real* які зміни відбудуться в програмі п.2 ?

4. Як зміниться процес вводу елементів масиву при заміні від *Write*(‘Элемент ‘,*s*,’=’) на *Write*(‘Элемент *s*=’)

Лабораторна робота №4

Підпрограми

Мета роботи : Дослідити особливості роботи з підпрограмою процедурою та підпрограмою функцією.

Теоретичні відомості.

Важливим принципом сучасного програмування – виступає принцип модульності. Це принцип структурованої програми, шляхом розбиття її на ряд самостійних фрагментів, зв'язаних з основною програмою лише декількома параметрами.

Function < ідентификатор > (< список параметрів >): < тип функції >.

function (функція) – зарезервованне слово. < Ідентифікатор > – ідентификатор функції. Значение цього ідентифікатора повертає підпрограма-функція. Тип ідентифікатора є <типом функції>.

<Список параметрів> – довільний набір ідентификаторів-параметрів, що передаються в функцію з вказівкою на їх тип. Однотипові параметри можуть передаватися групами. Групи отділені один від одного крапкою з комою. Список параметрів може бути відсутній (разом з дужками).

Procedure < ідентификатор > (<список параметрів >);

procedure (процедура) – зарезервованне слово.

< ідентификатор > – назва підпрограми.

< Список параметрів > – довільний набір ідентификаторів-параметрів, що передаються в процедуру з вказівкою на їх тип. Однотипові параметри можуть передаватися групами. Групи отділені один від одного крапкою з комою. Список параметрів може бути відсутній (разом з дужками).

Якщо стоїть необхідність в поверненні з процедури групи параметрів, необхідно встановити слово *var* перед відповідною групою.

Хід роботи.

1. Завантажити середовище *Pascal*.
2. Набрати текст програми по роботі підпрограми функції

   Program LR31;

```
Var a, b : integer;
        x, y : real;
function  Fx(x: real): real;
 begin
        Fx:=sin(x)+sqrt(x)
 end;
    Begin
    b:=20;  a:=5;
    writeln('x='); read(x);
    y:=Fx(x);
    writeln('y=',y)
End.
```

3. Занести результат роботи до протоколу

Y=_____.

4. Набрати текст програми по роботі підпрограми функції

```
Program LR31;
Var a, b : integer;
        x, y : real;
procedure  Fx(x: real; var y:real);
 begin
        y:=sin(x)+sqrt(x)
 end;
    Begin
    b:=20;  a:=5;
    writeln('x='); read(x);
    Fx(x,y)
    writeln('y=',y)
End.
```

5. Занести результат роботи до протоколу

Y=_____.

6. Написати програму по обрахунку значення *y* функції (підпрограма функція з п.2), в якій значення *x* змінюється в циклі з кроком 1 від 5 до 15. В кожному кроці циклу виводи значення *y*.

7. Занести значення *x* та *y* до протоколу

X:

_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

Y:

_____|_____|_____|_____|_____|_____|_____|_____|_____|_____|

Звіт повинен містити назву роботи, мету, короткі теоретичні відомості, тексти виконуваних програм п.2, 4, 6 та їх результати роботи.

Контрольні запитання

1. Навести приклад заголовку підпрограми-функції ?
2. Навести приклад заголовку підпрограми-процедури ?
3. Де в наступному рядку помилка ?

```
Function Fx (x:real; var y:real):integer;
```

4. Де в наступному рядку помилка ?

```
Procedure ABC(x:integer, y:real): real;
```

Лабораторна робота №5

Побудова графіків елементарних функцій.

Мета роботи : Дослідити графічний режим в *Pascal* на прикладі побудови графіка функції однієї змінної.

Теоретичні відомості

Підключаючи модуль *Graph* користувач може працювати в графічному режимі. В даній роботі відбувається побудова графіка функції, на прямокутну ділянку розміром *L* пікселів по горизонталі та *H* пікселів по вертикалі відображується прямокутна область площини *X*0*Y*, обмежена по горизонталі значеннями *Xmin*, *Xmax*, а по вертикалі, *Ymin* та *Ymax*, відповідно до заданого варіанту *Nvr*.

Визначення параметрів системи координат починаємо з обчислення *Dx* – ціни одного пікселя по горизонталі та *Dy* – по вертикалі, за умови, що ширина *L* та висота *H* заданої області на екрані використовуються повністю (це характерно для *God=false*).

Dx=(Xmax-Xmin)/L;

Dy=(Ymax-Ymin)/H.

Після визначення фактичних розмірів прямокутника *L*0 x *H*0, він розміщується в центрі екрана (координати *Xu*, *Yu* визначають лівий верхній кут прямокутника). Розраховуються також *X*0, *Y*0 – екранні координати точки, куди відображується початок системи координат X0Y. Для подальшої розмітки шкал, нанесення координатної сітки та оцифровування шкал визначаються початкові значення *Xn*, *Xk*, *Yn*, *Yk*.

Процедура *Clear* - призначена для стирання зображення в межах прямокутної області, яка задається координатами *X*1,*Y*1 лівого верхнього та *X*2,*Y*2 правого нижнього її кутів.

Горизонтальні та вертикальні прямі для зображення відповідних осей системи координат формує процедура *SystCoor*.

Формування на екрані ламаної, що зображує графік функції, заданої масивом *Mo*:*CoefL*, кольором *C* реалізується процедурою *Graphic* (*type CoefL* = *array*[-1..601] *of real*).

В нижньому рядку екрана розміщено інформаційний рядок, що інформує користувача про поточне значення змінних функції. Формується інформаційний рядок в змінній *Ts*:*String*[80], звільняє місце для неї і виводить *Ts* у відповідному місці процедура *Info*.

Модуль *Groms* відповідає за оцифровку та нумерацію графіка.

Хід роботи.

1. Завантажити середовище Турбо Паскаль.
2. Набрати текст програми по побудові графіків

```
Program Graphics;
    uses Crt, Graph, Serv;
    const Nvr: integer=1;
          A: real=-0/5;
          B: real=3;
          K: real=2;
     var  Mo: Coefl;
     function F(x: real): real;
        begin
          case Nvr of
             1: F: = sin(x);
             2: F: = cos(x);
             3: F: = cos(B*x)+A*sin(k*B*x);
             4: F: = 1-exp(A*x)*(cos(B*x)-A/B*sin(B*x))
          end
        end;
procedure Coefs;
    var T1: stringe; J1: char;
    begin
```

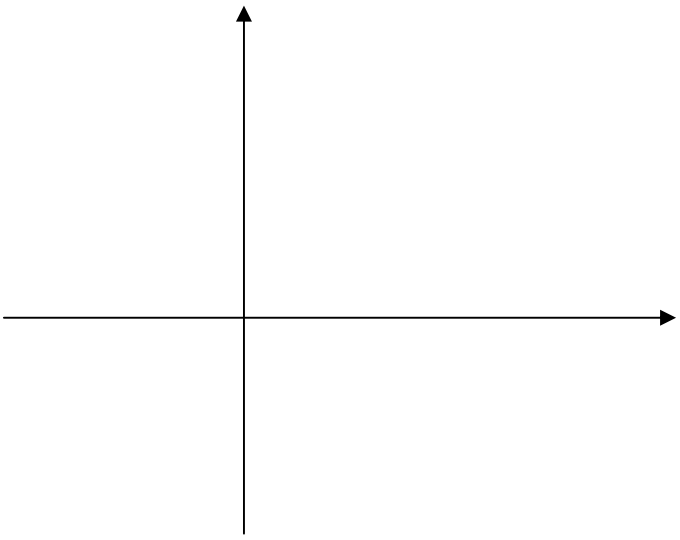
```
  repeat
    PutA; Ou('0-exit, 1-A, 2-B, 3-k');
    Str (A:1:3, T10); Ts:= 'A='+T10;
    Str (B:1:3, T10); Ts:= Ts+', B='+T10;
    Str (K:1:3, T10); Ts:= Ts+', K='+T10;
      case Nvr of
        1: T1: = 'sin(x)';
        2: T1: = 'cos(x)';
        3: T1: = 'cos(B*x)+A*sin(k*B*x)';
        4: T1: = 1-exp(A*x)*(cos(B*x)-A/B*sin(B*X))
      end;
      Ts: = Ts+'(F='+T1+')';
      Info; J1: = ReadKey;
      case J1 of
        '1': Our('A', A);
        '2': Our('B', B);
        '3': Our('K', K)
      end
  until J1='0'
end;
begin
  Xn: = 0; Xk: = 5;
  repeat
    PutA; Ou('Esc-exit,1-Nvr,2-Coefs,3-L,4-H,5-Xn,6-Xk,7-c,8-Sc,9- Gr');
    Str(Nvr, T10); Ts: = 'Nvr=' + T10;
    Str(L, T10); Ts: = Ts+', L=' + T10;
    Str(H, T10); Ts: = Ts+', H=' + T10;
    Str(Xn:1:3, T10); Ts: = Ts+', Xn=' + T10;
    Str(Xk:1:3, T10); Ts: = Ts+', Xk=' + T10;
    Str(c, T10); Ts: = Ts+', c=' + T10;
```

```
Info; J:= ReadKey;
case J of
   '1': Oui('Nvr(in[1..n])', Nvr);
   '2': Coefs;
   '3': Oui('L', L);
   '4': Oui('H', H);
   '5': Our('Xn', Xn);
   '6': Our('Xk', Xk);
   '7': Oui('c', c);
   '8': begin
          Xmin: = Xn; Xmax: = Xk;
          Ymin: = 0; Ymax: = 0;
          Dx: = (Xk-Xn)/L; Mo[-1]: = L;
          for S: = 0 to L do
            begin
              y: = F(Xn+S*Dx); Mo[S]: = y;
              if Y<Ymin then Ymin: = Y;
              if Y>Ymax then Ymax: = Y
            end:
          X0Y0(false); ClearDevice; SystCoor;
          Graphic(Mo, c)
        end;
   '9': begin
          Dx: = (Xk-Xn)/L; Mo[-1]: = L;
          for S: = 0 to L do
          Mo[S]: = F(Xn+S*Dx);
          Graphic(Mo, c)
        end
end
until J = #27;
```

CloseGraph

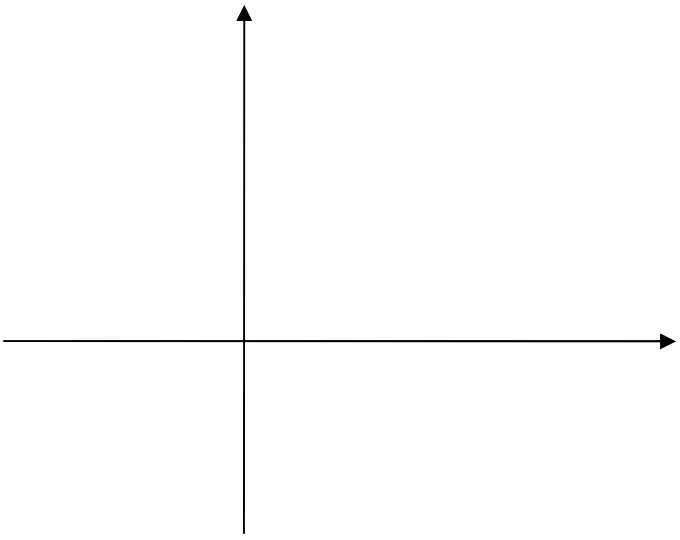
end.

3. Перенести в протокол результати побудови 4-х графіків(Nvr – 1, 2, 3,4)



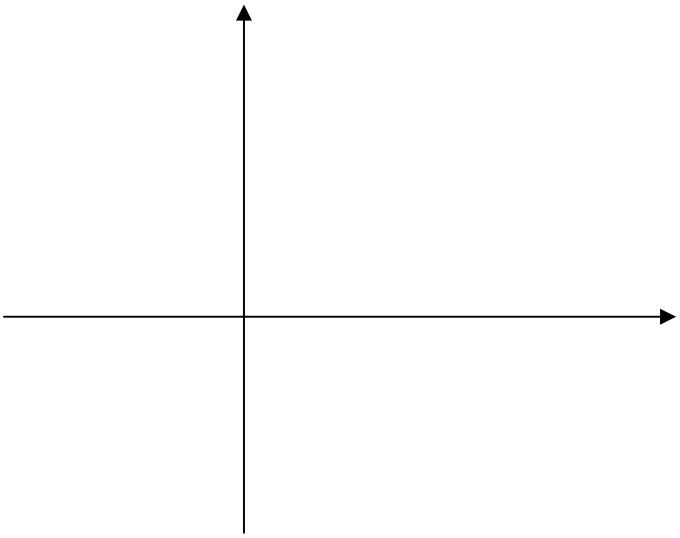
Nvr=1

F(x)=



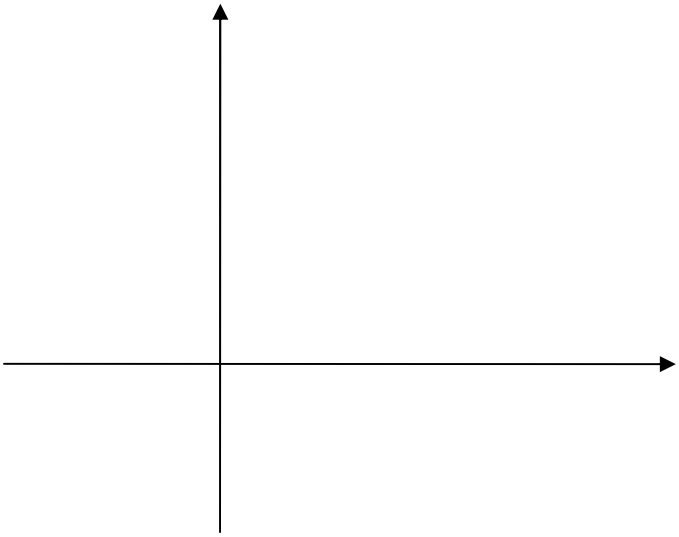
Nvr=2

F(x)=



Nvr=3

F(x)=



Nvr=4

F(x)=

Звіт повинен містити назву роботи, мету, короткі теоретичні відомості, тексти виконуваної програми п.3, результати побудови графіків в залежності від значення параметра *Nvr*.

Контрольні запитання

1. Пояснити механізм формування інформаційного рядка в процедурі *Coefs*.
2. Що відбувається в рядку *X0Y0*(*false*); *ClearDevice*; *SystCoor*; *Graphic*(*Mo*, *c*) ?
3. Що відбувається в 9-му пункті основного меню програми *Graphics* ?

Лабораторна робота №6

Побудова годографів.

Мета роботи : Дослідити побудову функціональних залежностей заданих параметрично, використовуючи графічний режим в *Pascal*.

Теоретичні відомості

Годографи – це графічне зображення залежностей між двома змінними, що задані параметрично – $X=f1(\omega)$ та $Y=f2(\omega)$. $f1(\omega)$ та $f2(\omega)$ – деякі функції параметра ω. У загальному випадку ці функції можуть задаватись і неявно, наприклад шляхом реалізації якого завгодно алгоритму. В подальшому будемо орієнтуватися на те, що глобальні змінні *X*, *Y*: *real* (визначені в *Serv*) отримують свої значення в функції від поточного значення параметра *w*: *real* (програмний еквівалент параметра ω).

В роботі розглянуто 5 варіантів побудови годографів відповідно до *Nvr*.

Nvr=1. Це коло радіуса *R*0 (для зручності вважаємо, що *R*0 задається в пікселях), звідси і R0:integer.

Nvr=2. Це прямий еліпс (з осями паралельними осям системи координат *X*0*Y*). Півосі еліпса – R1 (по горизонталі) та R0 (по вертикалі).

Nvr=3. Це гіпоциклоїда, у якої *R*0- радіус нерухомого кола, *R*1 – радіус колеса, що котиться всередині нерухомого без ковзання, *U* – довжина штанги, закріпленої на рухомому колесі (точніше – це відстань від центра рухомого колеса до точки фіксації пера на штанзі, саме це перо і малює гіпоциклоїду).

Nvr=4. Це епіциклоїда. Тут рухоме колесо котиться зовні по нерухомому.

Nvr=5. Нахилений еліпс, де *U* – кут повороту еліпса навколо його осі. Кути в Турбо Паскалі прийнято задавати в градусах, саме тому *U*: *integer*.

Параметр *w* це кут (в радіанах) повороту радіуса-вектора для *Nvr*=1,2,5 та кут нахилу лінії, що з'єднує осі нерухомого та рухомого коліс для *Nvr*=3,4. За центр усіх перерахованих фігур приймається початок координат (X0,Y0).

Для визначення *Xmin*, *Xmax*, *Ymin*, *Ymax* будемо сканувати діапазон *Wn*..*Wk* з кроком *Dw* := (*Wk*-*Wn*)/*Nsc*, де *Nsc* (глобальний параметр типу

integer або *word*) – кількість кроків у межах діапазону. Тут варто зробити таке зауваження. Існують годографи, на яких мітки значень параметра розподіляються надзвичайно нерівномірно (ну, наприклад, грубо кажучи, перша половина годографу відповідає діапазону 0..0,1, а друга – 0,1..∞).

Механізм функціонування підпрограми *Scan* :

Обчислюється значення глобальної змінної *Xu*:= (*GetMaxX*-*Nsc*) *div* 2;

Початкові значення *Xmin*, *Xmax*, *Ymin*, *Ymax* перед початком сканування приймають рівними нулю;

Завершує підпрограму процедура *X*0*Y*0(*true*), яка визначає параметри системи координат *X*0*Y*.

Процедура *PointGod*, домальовує черговий піксель годографа.

Процедура *Godo*, "керує" формуванням годографа в цілому.

Хід роботи.

1. Завантажити середовище Турбо Паскаль.
2. Набрати текст програми по побудові гродографів

```
program  Godograf;
    uses Crt, Graph, Serv;
    const Nvr: integer=1;
          R0: integer=50;
          R1: integer=20;
          U: integer=10;
          Nsc: integer=100;
          Wn: real=0;
          Wk: real=6.28;
    var W, Dw:real;
   procedure UrGod;
 var Xe, Ye, Ur: real;
  begin
    case Nvr of
      1: begin
```

```
            X:= R0*cos(w); Y:= R0*sin(w)
          end;
        2: begin
            X:= R1*cos(w); Y:= R0*sin(w)
          end;
        3: begin
            Xe:= R0-R1; Ye:= Xe*w/R1;
            X:= Xe*cos(w)-U*cos(Ye);
            Y:= Xe*sin(w)-U*sin(Ye)
          end;
        4: begin
            Xe:= R0+R1; Ye:= Xe*w/R1;
            X:= Xe*cos(w)-U*cos(Ye);
            Y:= Xe*sin(w)-U*sin(Ye)
          end;
        5: begin Ur:= U*Pi/180;
            Xe:= R1*cos(w); Ye:= R0*sin(w);
            X:= Xe*cos(Ur)-Ye*sin(Ur);
            Y:= Ye*cos(Ur)+Xe*sin(Ur)
          end
        end
    end;
procedure Scan;
  var S:integer;
      J:char;
  begin
    Dw:= (Wk-Wn)/Nsc;
    Xu:= (GetMaxX-Nsc) div 2;
    PutA; Rectangle (Xu, 0, Xu+Nsc, 5);
    Ou ('0-exit'); S:=0; J:='1';
```

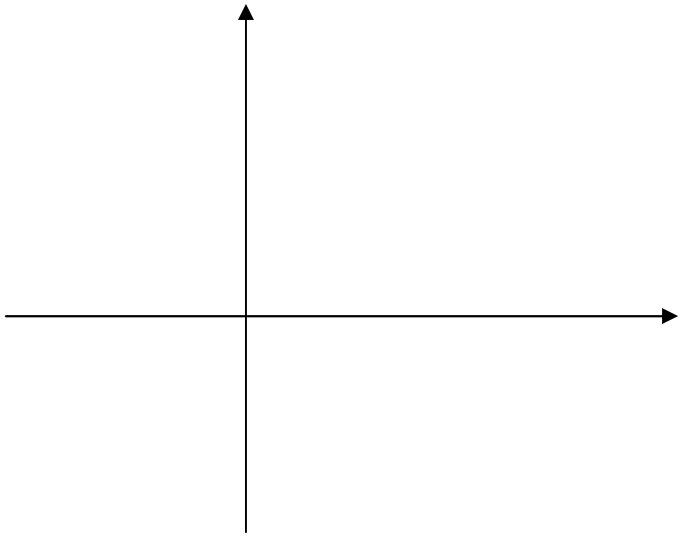
```
    Xmin:= 0; Xmax:= 0; Ymin:= 0; Ymax:= 0;
    repeat
      W:= Wn+S*Dw; UrGod;
      Line (Xu+s, 1, Xu+S, L);
      If X<Xmin then Xmin:=X;
      If X>Xmax then Xmax:=X;
      If Y<Ymin then Ymin:=Y;
      If Y>Ymax then Ymax:=Y;
      inc(S);
      if KeyPressed then J:=ReadKey
    until (S>Nsc) or (J='0');
    X0Y0 (true)
  end;
    Procedure SZ;
      begin
        UrGod;
        if Dx<>0 then S:=X0+round (X/Dx)
                 else  S:=X0;
        if Dy<>0 then Z: =Y0-round (Y/Dy)
                 else Z: =Y0
      end;
Procedure  PointGod;
      const Kw=1.2;
      var Mdx, Mdy, Md, Sp, Zp:integer;
          Wp:real;
      begin
        Wp:=W; Sp:=S; Zp:=Z;
        repeat
          W:=Wp+Dw: SZ;
          Mdx:=abs (S-Sp); Mdy:=abs (Z-Zp);
```

```
          if Mdx>Mdy then Md:=Mdx
                          else Md:=Mdy;
          if Md<>1 then
             if Md>1 then Dw:=Dw/Md
                     else Dw:=Dw*Kw

       until (Md=1) or (W>Wk);
       if Md=1 then
       if C>=0 then PutPixel (S,Z,C)
                else PutPixel (S,Z,GetMaxColor-GetPixel (S,Z))
    end;
Procedure LineGod;
   var Mdx, Mdy, Md, Sp, Zp:integer,
       Kw, Sr, Wp:real,
   begin
     Wp:=W; Sp:=S; Zp:=Z;
      Sr:=(Min+Max)/2; Kw:=Max/Min/2;
      repeat
         W:=Wp+Dw; SZ;
         Mdx:=abs (S-Sp); Mdy:=abs (Z-Zp);
         if  Mdx>Mdy then Md:=Mdx
                          else Md:=Mdu;
         if (Md>Max) or (Md<Min) then
         if Md>Max then Dw:=Dw*Sr/Md
                   else Dw:=Dw*Kw
      until ((Md<=Max) and (Md>=Min)) or (W>Wk);
      if (Md<=Max) and (Md>=Min) then
         begin
           Line (Sp, Zp, S, Z);
           If C<0 then PutPixel (S, Z, GetMaxColor-GetPixel (S, Z))
```

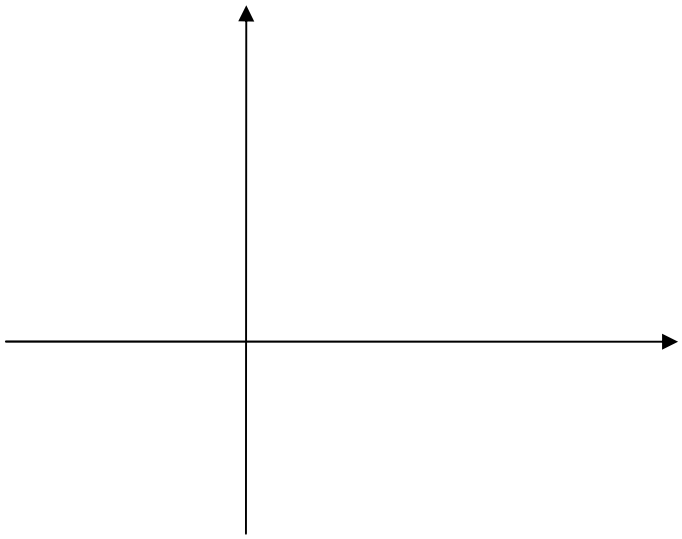
```
        end
end;
procedure  Godo;
   var J:char;
   begin
     Dw:=(Wk-Wn)/Nsc; W:=Wn; SZ;
     if C>=0
        then
          begin
            SetColor (C); PutPixel (S, Z, C)
          end
        else
          SetWriteMode (1);
          case Tg of
            'P': PutPixel (S,Z)
            'L': SetWriteMode (1)
          end;
     PutA; J:='1'; Ou ('0-exit');
     OutTextXY (434, 0, 'W=');
     repeat
       case Tg of
         'P': PointGod;
         'L': LineGod
       end;
       Str (W:1:3, T10); Clear (450, 0, GetMaxX-1, 12);
       OutTextXY (450, 0, T10);
       If KeyPressed then J:= ReadKey
     until (W>Wk) or (J='0');
     if C>=0 then SetColor (15)
                 else SetWriteMode(0)
```

```
        end;

Begin
  repeat
    PutA; Ou ('Esc-exit, 1-Nvr, 2-R0, 3-R1, 4-U, 5-Wn, 6-Wk, 7-C, 8-Scan,
                                                          9-Gdg, 0-Sc');
    Str (Nvr (Nvr, T10);  Ts:= 'Nvr='+T10;
    Str (R0, T10);  Ts:= Ts+', R0='+T10;
    Str (R1, T10);  Ts:= Ts+', R1='+T10;
    Str (U, T10);  Ts:= Ts+', U='+T10;
    Str (Wn:1:2, T10);  Ts:= Ts+', Wn='+T10;
    Str (Wk:1:2, T10);  Ts:= Ts+', Wk='+T10;
    Str (C, T10);  Ts:= Ts+', C='+T10;
    Info; J:= ReadKey;
    Case J of
      '1': Oui ('Nvr', Nvr);
      '2': Oui ('R0', R0);
      '3': Oui ('R1', R1);
      '4': Oui ('U', U);
      '5': Our ('Wn', Wn);
      '6': Our ('Wk', Wk);
      '7': Oui ('C', C);
      '8': Scan;
      '9': Godo;
      '0': SystCoor
    end
  until J=#27;
  CloseGraph
end.
```

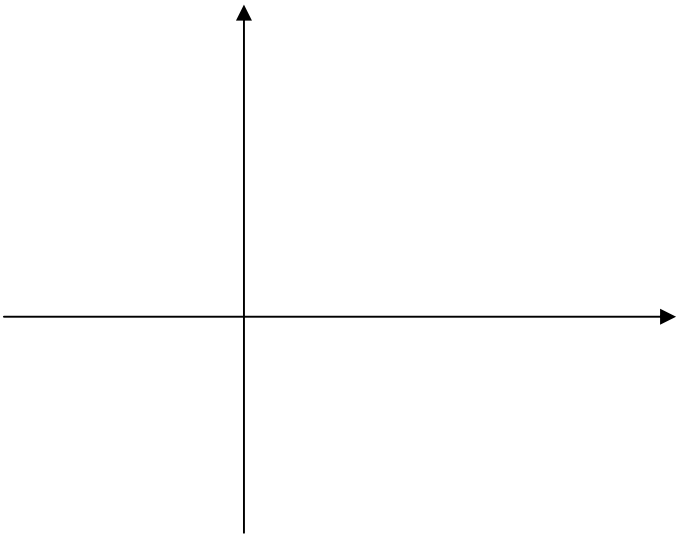
3. Перенести в протокол результати побудови 5-ти годографів (*Nvr* – 1, 2, 3, 4, 5)



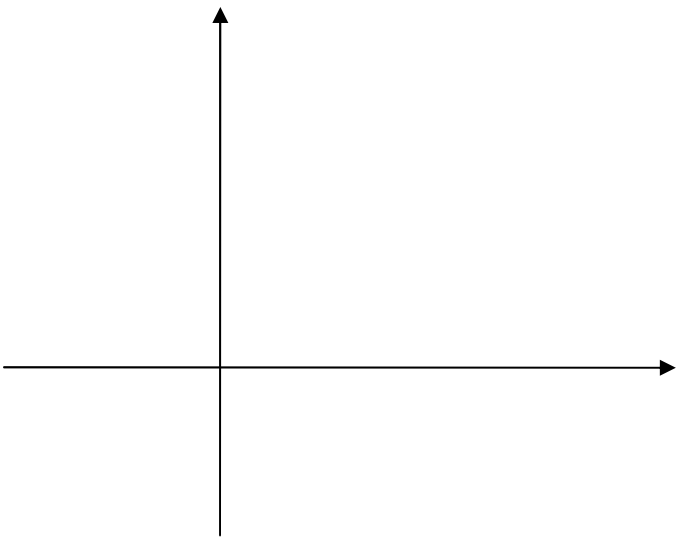
Nvr=1



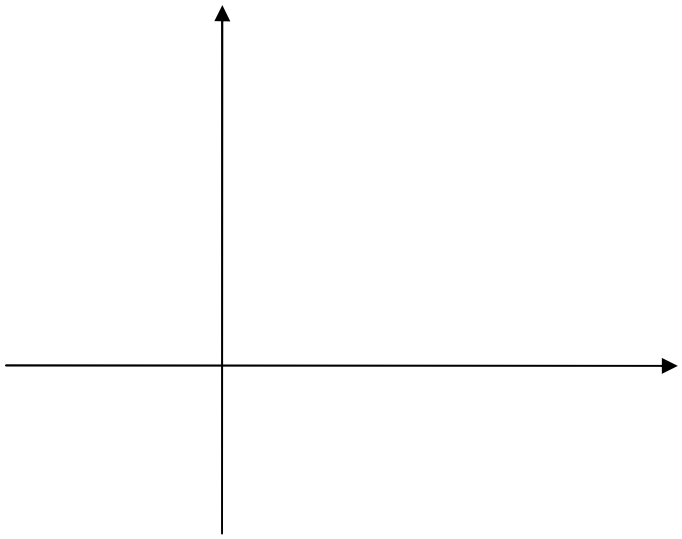
Nvr=2



Nvr=3



Nvr=4



Nvr=5

Звіт повинен містити назву роботи, мету, короткі теоретичні відомості, тексти виконуваної програми п.2, результати побудови годографів в залежності від значення параметра *Nvr*.

Контрольні запитання

1. Пояснити механізм роботи процедури *UrGod*.
2. Пояснити механізм роботи процедури *Scan*.
3. Пояснити механізм роботи процедури *SZ*.
4. Пояснити механізм роботи процедури *PointGod*.
5. Пояснити механізм роботи процедури *Godo*.

Лабораторна робота №7

Побудова системи ізоліній.

Мета роботи : Дослідити побудову ліній одного рівня - ізоліній, використовуючи графічний режим в *Pascal*.

Теоретичні відомості

До цього часу мова йшла щодо графічних зображень залежностей між *x* та *y*, заданих явно (у вигляді функцій) або параметрично, тобто зображувалась функція одного параметра (*x* або ω).

Функцію f(x,y) можна уявляти собі у вигляді поверхні, сформованої над заданою ділянкою площини *X*0*Y*. Отримати уявлення про цю поверхню можна проектуючи на площину хоу перетини цієї поверхні площинами , перпендикулярними осі *z* (вздовж якої відкладається значення функції *f*). Лінії перетину зображуваної поверхні такими площинами характерні тим, що значення функції у кожній точці даної лінії одне й те ж, звідси й назва – ізоліній (лінії однакових значень функції). Таким чином, кожній ізолінії ставиться у відповідність певне значення функції (індекс ізолінії). Сукупність ізолінії для ряду значень функції (з позначенням рівнів на них) дає чітке уявлення про характер поверхні. Цим прийомом користуються картографи (особливо військові) – їх карти покритті лініями рівних висот (або глибин) відносно рівня моря – це найчистішої води ізолінії.

Задана ділянка площини *X*0*Y* (наприклад, прямокутна розмірами (*Xmax*-*Xmin*)*(*Ymax*-*Ymin*)) відображується на екрані комп'ютера у вигляді прямокутника *L* x *H*. Для поточного пікселя екранної області обчислюємо значення функції *F* та визначаємо *Nrr*:= *F*/*Hiz*. Тут *Nrr* – номер рівня.

Хід роботи

1. Завантажити середовище Турбо Паскаль.
2. Набрати текст програми по побудові ізоліній

```
program Izolines;
uses crt,graph,serv;
const Hiz: real=0.2;
```

```
      Mi: integer=30000;
      Nvr: integer=1;
var a: coef;
procedure HorComp(a: Coef; R,I: real; var Re,Im: real);
var R1: real;
    n,s: integer;
begin
 n:=round(a[-1]); Re:=a[n]; Im:=0;

 For s:=n-1 downto 0 do
  begin
   R1:=Re*R-Im*I+a[s];
   Im:=Re*I+Im*R;
   Re:=R1;
  end;
end;
Function Fxy(x,y: real):real;
var Re,Im: real;
begin
 case Nvr of
      1: Begin
          HorComp(a,x,y,Re,Im);
          Fxy:=sqrt(sqr(Re)+sqr(Im));
         end;
      2: Fxy:=sqr(sin(x))+sqr(cos(y));
      3: Fxy:=1-cos(Pi*x)/2+sqr(y);
 end;
end;
procedure Izo;
var Nr,s,z: integer;
```

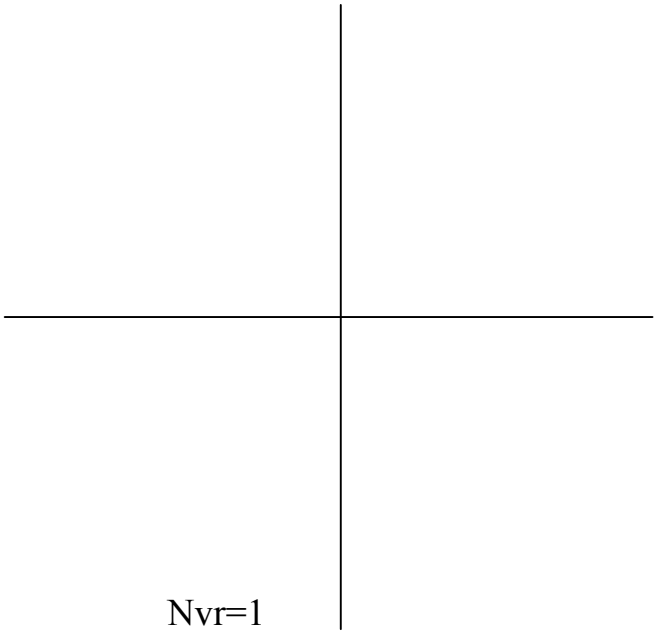
```
    Nrr: real;
    Mnr: array[-1..800]of integer;
begin
 for z:=-1 to H0 do
    begin
      Nrr:=Fxy(Xmin-Dx,Ymin+Z*Dy)/Hiz;
      if abs(Nrr)>Mi then  mNr[z]:=Mi
                else mnr[z]:=trunc(Nrr);
    end;
  for s :=0 to L0 do
    begin
       x:=xmin+s*dx;
       for z:=-1 to H0 do
         begin
           nrr:=fxy(x,ymin + z *dy)/hiz;
           If abs(nrr)>Mi then Nr:=Mi
                       else Nr:=trunc(Nrr);
           If z>-1 then
             begin
               if (Nr<>Mnr[z-1])or(Nr<>Mnr[z]) then
                   if abs(Nrr)<Mi then
                      PutPixel(Xu+s,Yu+H0-z,{round(abs(nrr))mod 16} 15);
              if abs(Nrr)<1 then
                  if (Nrr*Mnr[z-1]<0)or(Nrr*Mnr[z]<0) then
                     PutPixel(xu+s,yu+h0-z,round(abs(Nrr)mod 16);

             end;
           Mnr[z]:=Nr;
        end;
        end;
```

```
end;
Function Radius(A: coef): real;
var s,n: integer;
    b: real;
begin
 n:=round(A[-1]);
 b:=abs(a[0]);
 for s:=1 to n-1 do
  if abs(a[s])>b then b:=abs(a[s]);
 radius:=1+b/(abs(a[n]));
end;
procedure Diap;
begin
 repeat
  PutA; Ou('0-exit,1-Xmin,2-xmax,3-Ymin,4-Ymax');
  if Nvr=1 then
   begin
    x:=Radius(a);
    str(x:1:3,t10);
    Ts:='Radius='+t10+' ';
   end
          else Ts:=' ';
  str(Xmin:1:3,T10); Ts:=Ts+',Xmin='+T10;
  str(Xmax:1:3,T10); Ts:=Ts+', Xmax='+T10;
  str(Ymin:1:3,T10); Ts:=Ts+', Ymin='+T10;
  str(Ymax:1:3,T10); Ts:=Ts+', Ymax='+T10;
  info; j:=readkey;
  case j of
   '1': Our('Xmin',Xmin);
   '2': Our('Xmax',Xmax);
```

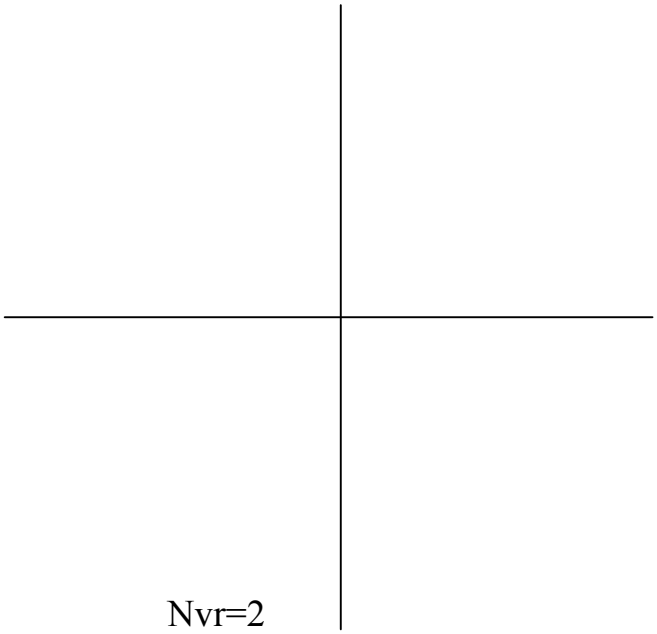
```
   '3': Our('Ymin',Ymin);
   '4': Our('Ymax',Ymax);
  end;
 until j='0';
end;
Procedure InpPol(Id: string; var A: Coef);
Var s, n: Integer;
Begin
 Oui('n-step pol',n);
 A[-1] := n;
 For s := 0 to n do
  Begin
   Str(s, T10);
   Our(Id+'['+T10+']',A[s])
  end
End;
begin
 Xmin:=-2; Xmax:=2; Ymin:=-2; Ymax:=2; c:=15;
 A[-1]:=5; A[0]:=1; a[5]:=1;
 for s:=1 to 4 do A[s]:=0;
 repeat
  PutA; Ou('Esc-exit,1-Nvr,2-InpPol,3-L,4-H,5-Diap,6-Hiz,7-Clear,8-SC,9-
Izo');
  Str(Nvr,t10); Ts:='Nvr='+t10;
  Str(L,t10); Ts:=Ts+', L='+t10;
  Str(H,t10); Ts:=Ts+', H='+t10;
  Str(Hiz:1:2,t10); Ts:=Ts+', Hiz='+t10;
  Info; J:=readkey;
  case J of
   '1': oui('Nvr',nvr);
```

```
  '2': InpPol('A',A);
  '3': oui('L', L);
  '4': oui('H',h);
  '5': Diap;
  '6': Our('Hiz',Hiz);
  '7': Cleardevice;
  '8': begin
         X0Y0(true);
         systcoor;
       end;
  '9': Izo;
 end;
until J=#27;
end.
```

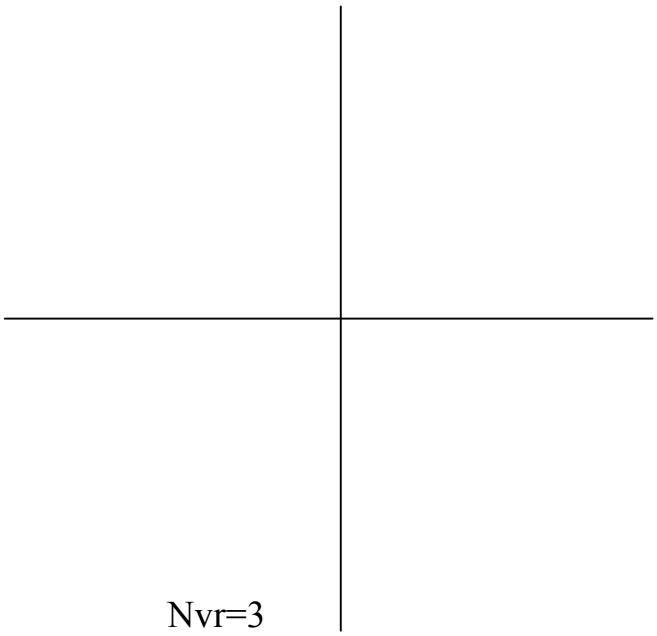
3. Перенести в протокол результати побудови 3-х варіантів (*Nvr*)



Nvr=1



Nvr=2



Nvr=3

Звіт повинен містити назву роботи, мету, короткі теоретичні відомості, тексти виконуваної програми п.2, результати побудови ізоліній в залежності від значення параметра *Nvr*.

Контрольні запитання

1. Пояснити механізм роботи процедури *HorComp*.
2. Пояснити механізм роботи процедури *InpPol*.
3. Пояснити механізм роботи функції *Radius*.
4. Пояснити механізм роботи процедури *Izo*.

Список рекомендованої літератури.

1. Аверіна Т.В., Кубрак Н.А. Динаміка елементів систем : Навч. посібник – К.: ІЗМН, 1998 – 224 с.
2. Карачун В.В., Кваско М.З., Кубрак Н.А. Прикладний аналіз і візуалізація характеристик динамічних систем : Навч. посібник – К.: ІЗМН, 1999 – 138 с.
3. Кубрак Н.А. Хвильові процеси в гнучких ланках автоматичних систем : Навч. посібник – К.: НМЦ ВО, 2000 – 160 с.
4. Калиткин Н.Н. Численные методы – М.: Наука, 1978 – 512 с.
5. Кошляков Н.С., Глинер Э.Б., Смирнов М.М. Основные дифференциальные уравнения математической физики – М.: Физмат, 1962 – 767 с.
6. Кубрак А.І. Ідентифікація динамічних характеристик елементів систем керування. Част. 1. Математичні методи : Навч. посібник – К.: ІСДО, 1995 – 208 с.