УДК 515.2

Ткаченко В.П.,
Челомбітько В.Ф.,
Харківський національний університет радіоелектроніки, Україна

РЕАЛІЗАЦІЯ ДЕЯКИХ ПРИНЦИПІВ ПОБУДОВИ ВІЗЕРУНКІВ ЗАСОБАМИ ОПЕРАТОРІВ МОВИ MAPLE

Анотація: розглянуто принципи складання та побудови орнаментів у середовищі алгоритмічної мови Maple.

Ключові слова: візерунок, геометричні перетворення, рекурсивні побудови, maple

Постановка проблеми. Орнамент – це візерунок, що складається з ритмічно впорядкованих елементів. Орнамент будується за особливими законами, за допомогою певних засобів. Головний засіб – це геометричне перетворення. Тому актуальними будуть дослідження, спрямовані на формалізацію побудови засобами обчислювальної техніки орнаментів, реалізованих засобами операторів алгоритмічної мови, наприклад, Maple.

Аналіз останніх досліджень та публікацій. Геометричне перетворення площини – це взаємнооднозначне відображення цієї площини на себе. Найбільш важливим геометричним перетворенням є рух, тобто перетворення, що зберігає відстань, а також масштабування [1–3]. Прикладами руху є центральна симетрія, паралельний перенос, поворот і осьова симетрія. Мотив – це головний елемент орнаменту, основна його складова. Подальшим узагальненням буде побудова орнаментів за допомогою комп’ютерних технологій.

Формулювання цілей статті. Визначення можливостей реалізації головних принципів побудови візерунків у певному середовищі алгоритмічної мови.

Основна частина. Графічні візерунки звичайно створюють для декоративних заставок, реклами, демонстрації можливостей апаратури. Програми для створення комп'ютерних візерунків звичайно базуються на досить простих алгоритмах, які можна вважати тестами майстерності програмування. Головним тут є ідея створення декоративного ефекту.

Постановка завдання. Розробити алгоритми складання та побудови орнаментів на площині у середовищі алгоритмічної мови Maple, які базуються на головних принципах створення візерунків.

Основна частина. Якщо фігуру переміщати не обертаючи відносно свого "центра", то одержимо плоско–паралельний рух фігури, коли будь–який відрізок прямої на фігурі залишається паралельним самому собі. За "центр" фігури може бути прийнята будь–яка точка жорстко пов'язана з фігурою.

Звичайно "центр" фігури (xf, yf) переміщають відносно центра візерунка (xc, yc) за певним законом xf = xc + Fx; yf = yc + Fy, де Fx, Fy – функції від параметрів. У загальному випадку фігура може переміщатися, обертаючись відносно свого "центра" і деформуватися. При цьому параметри процедури зображення фігури повинні включати всі координати точок, які з'єднуються лініями. Координати і–тої точки фігури визначаються за формулами:

xxi := xf + Kxi * ((xi–xf) * cos(A) – (yi–yf) * sin(A)),
yyi := yf + Kyi * ((yi–yf) * cos(A) + (xi–xf) * sin(A)),

де *A* – кут повороту фігури відносно свого "центра", відлічуваного у лівій системі координат екрана за годинниковою стрілкою стосовно осі *X*,

xi, yi – початкові координати і–тої точки фігури,

xxi, yyi – нові координати і–тої точки фігури,

Kxi, Kyi – коефіцієнти масштабування координат і–тої точки по осях *X і Y.*

Приведемо приклад Maple – програми побудови візерунка, у якій реалізовано множину законів руху ліній відносно свого "центра".

```
q := evalf(W);
m := [[0,0],[1,0],[cos(q), sin(q)],[0,0]]:
w := []: t := T:
for k to N do
w := [op(w), m]:
sm := [m[2], m[3], m[1], m[2]]:
m := (1–t)*m + t*sm:
od:
```

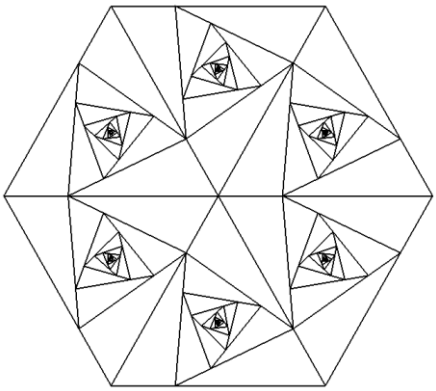
Після наведеного підготовчого блоку слід скласти шість блоків, які відрізнятимуться коефіцієнтами у парах тригонометричних функцій

```
a1 := plot(w, axes=none, scaling=constrained):
m := [[0,0],[cos(2*q),sin(2*q)], [cos(1*q),sin(1*q)],[0,0]]:
w := []: t := T:
for k to N do
w := [op(w),m]:
sm := [m[2],m[3],m[1],m[2]]:
m := (1–t)*m+t*sm:
od:
```

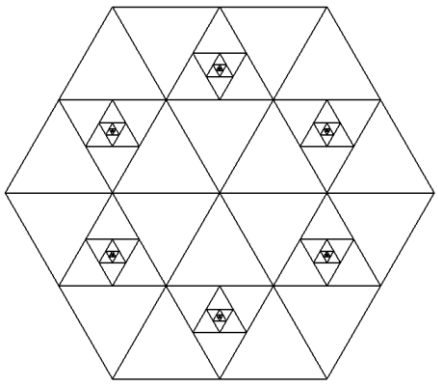
В результаті виконання всіх шести блоків матимемо заготовку для побудови зображення з використанням оператора

display(a1,a2,a3,a4,a5,a6, thickness=2);

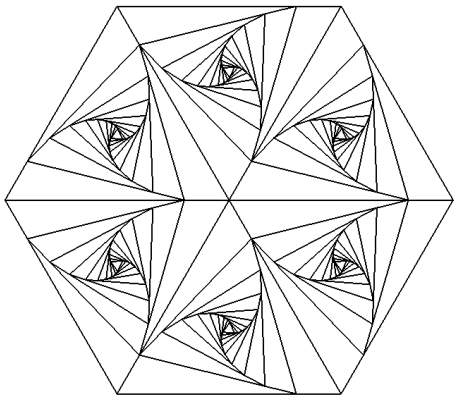
На рис. 1 наведено при **N = 10** приклади виконання програми побудови візерунка залежно від параметрів **W** і **T**.



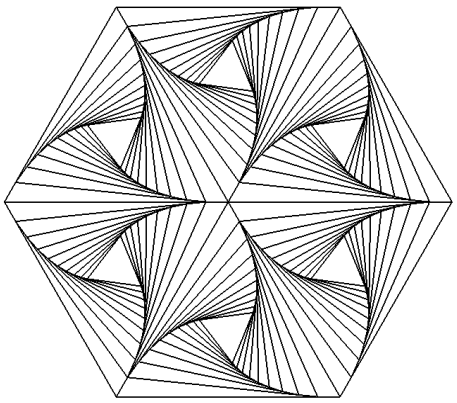
T = 0.3; W = Pi/3



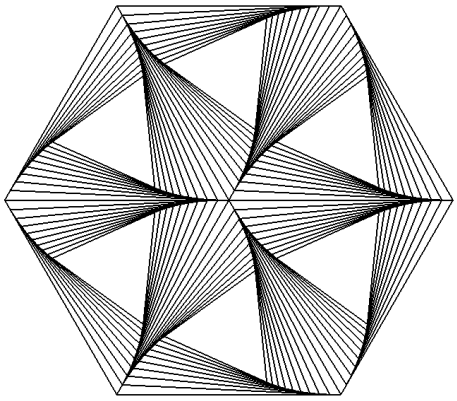
T = 0.5; W = Pi/3



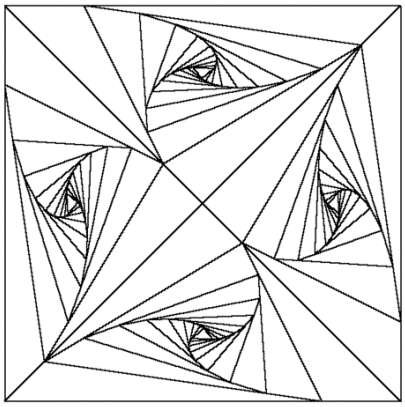
T = 0.8; W = Pi/3



T = 0.9; W = Pi/3



T = 0.95; W = Pi/3



T = 0.2; W = Pi/2

*Рис. 1. Приклади виконання програми побудови візерунка залежно від параметрів **T** і **W** .*

У програмуванні часто використовують рекурсивні оператори, наприклад, процедури, які містять звертання до самих себе. Такі звертання можуть бути прямими – тобто викликом процедури усередині самої процедури, або непрямими – викликом інших процедур, усередині яких є виклик вихідної процедури.

Наведемо приклад програми рекурсивної побудови візерунка, де процедура **drevo** звертається до тієї ж процедури **drevo**.

```
drevo := proc(L, N, x0, y0)
local i; global s, p;
```

```
options remember;
s := s+1;
p[s] := plot([x0 + L*sqrt(M)*sin(M*t)*cos(t),
y0 + L*sqrt(M)*sin(M*t)*sin(t), t=–W..W]):
if N > 1 then drevo(L/2, N–1, x0–L, y0+L);
drevo(L/2, N–1, x0+L, y0+L) fi;
RETURN(display([seq(p[i], i=1..2^N–1)], axes=NONE,
scaling=constrained, numpoints=1000,thickness=2))
end:
s:=0: drevo(100, 7, 0, 0);
```

На рис. 2 наведено приклади виконання програми рекурсивної побудови візерунка залежно від параметрів **W** і **M**.

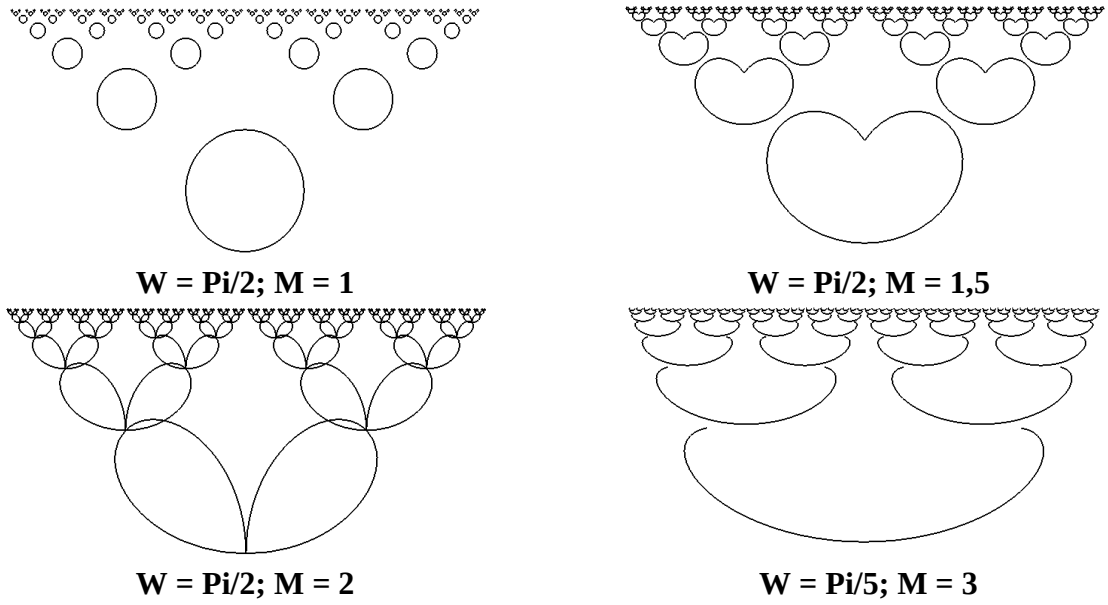


*Рис. 2. Приклади виконання програми рекурсивної побудови візерунка залежно від параметрів **W** і **M**.*

Цікавим є створення візерунків побудовою дзеркальних відображень фігури. В цьому випадку принцип побудови візерунків близький до зображень дзеркального калейдоскопа. У калейдоскопі система із трьох дзеркал створює ефект декількох шестиразових відбиттів набору кольорових кристалів. Математично такий принцип побудови візерунка можна описати так. Є три вихідних з однієї точки променів – осей симетрії. Кут між променями дорівнює $2*\pi/3$. Будується перша (початкова) фігура в секторі між першим і другим променями. Потім будується друга фігура як дзеркальне відображення першої фігури відносно другого променя, далі третя фігура, як дзеркальне відображення другої фігури відносно третього променя й так далі.

Наведемо приклад програми створення візерунка побудовою відбиттів початкової фігури відносно трьох осей відбиття.

```
fosi1 := x*sin(Pi/3) – y*cos(Pi/3);
```

fosi2 := y;

fosi3 := x*sin(Pi/3) + y*cos(Pi/3),

де кут вимірюється від осі Ox проти годинникової стрілки.

Початкову фігуру (наприклад, квадрат) задамо рівнянням у вигляді

f := (x,y) –> a – abs(x–x0) – abs(y– y0):

Для побудови результату симетричного відбиття відносно першої, другої та третьої осі симетрії слід застосувати формули:

X1 := (x,y) –> y*sin(2*Pi/3)+ x*cos(2*Pi/3);

Y1 := (x,y) –> x*sin(2*Pi/3)– y*cos(2*Pi/3);

f1 := (x,y) –> f(X1, Y1);

X2 := x;

Y2 := –y;

f2 := (x,y) –> f1(X2, Y2);

X3 := (x,y) –> y*sin(–2*Pi/3)+ x*cos(–2*Pi/3);

Y3 := (x,y) –> x*sin(–2*Pi/3)– y*cos(–2*Pi/3);

f3 := (x,y) –> f2(X3, Y3);

Побудову при **a = 1,5; x0 = 2; y0 = 1** проміжних результатів відбиття здійснюємо операторами

F := implicitplot(f(x,y), x=–3..4, y=–3..4);

F1 := implicitplot(f1(x,y), x=–3..4, y=–3..4);

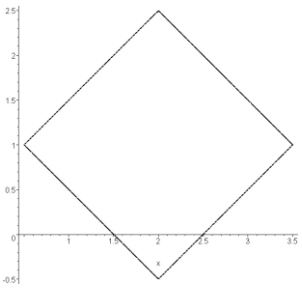
F2 := implicitplot(f2(x,y), x=–3..4, y=–3..4);

F3 := implicitplot(f3(x,y), x=–3..4, y=–3..4);

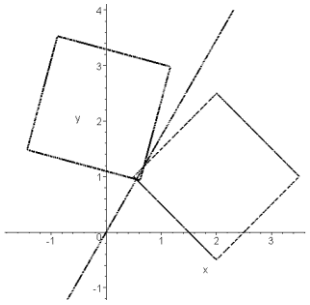
Результуюче зображення (яке можна продовжити в циклі) будується за допомогою оператора

display(F, F1, F2, F3 , thickness=3);

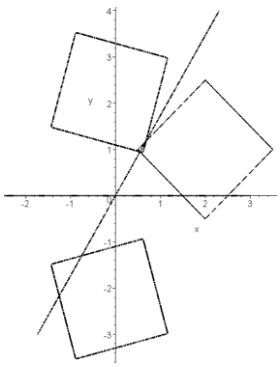
На рис. 3 наведено приклади побудованих зображень.



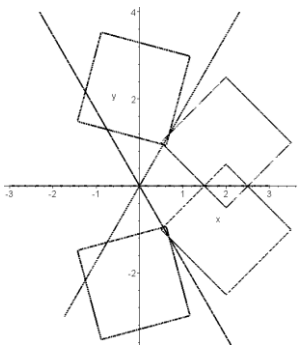
а) початкове зображення



б) результат першого відбиття



в) результат другого відбиття



г) результат третього відбиття

Рис. 3. Побудова симетричних зображень дзеркального калейдоскопа

Висновки. Головні принципи побудови візерунків можна реалізувати у середовищі алгоритмічної мови шляхом створення алгоритмів побудови зображень на площині.

Перспективи подальшого дослідження. Подальше дослідження принципів побудови візерунків.

Література

1. Классификация орнаментов – http://TMN.FIO.ru/works/80x/311/classif.htm
2. http:// graphfunk.narod.ru/ «Графики функций».
3. Котов Ю.В. Как рисует машина. – М.: Наука. 1988. – 224 с.

Аннотация

Ткаченко В.Ф., Челомбітько В.Ф. Реализация некоторых принципов построения узоров средствами операторов языка MAPLE. Рассмотрены принципы составления и построения орнаментов в среде алгоритмического языка Maple.

Ключевые слова: орнамент, геометрические преобразования, рекурсивные построения, MAPLE.

Abstract

Tkachenko V.P., Chelombitko V.F. Realization of some principles for drafting decorative patterns by facilities of operators of MAPLE. Principles of drafting and construction of decorative patterns are considered in the environment of algorithmic language of Maple.

Key words: decorative pattern, geometrical transformations, recursive constructions, MAPLE.