Temă: mulţimi Fatou, mulţimi Julia

1. JuliaGreen(). Să considerăm funcţia JuliaGreen() dată de exemplu la curs pentru metoda locului final şi să schimbăm numărul de iteraţii `nrIter = 100` în `nrIter = 101`.

```
def JuliaGreen():
    c = complex(0.45, 0.2)
    rhoMax = 1.0e20

    def f(z):
        # f(z)=(z*z*z+c)/(z*z*z-c)
        u = z * z * z
        if u == c:
            return rhoMax
        else:
            return (u + c) / (u - c)

    C.setXminXmaxYminYmax(-2.7, 3.5, -3.1, 3.1)
    nrIter = 100
    for coloana in C.screenColumns():
        for zeta in coloana:
            z = zeta
            for k in range(nrIter):
                z = f(z)
                if C.rho(z) >= rhoMax: break
            C.setPixel(zeta, Color.Index(sum(C.getHK(z))))
        if C.mustClose():
            return
```

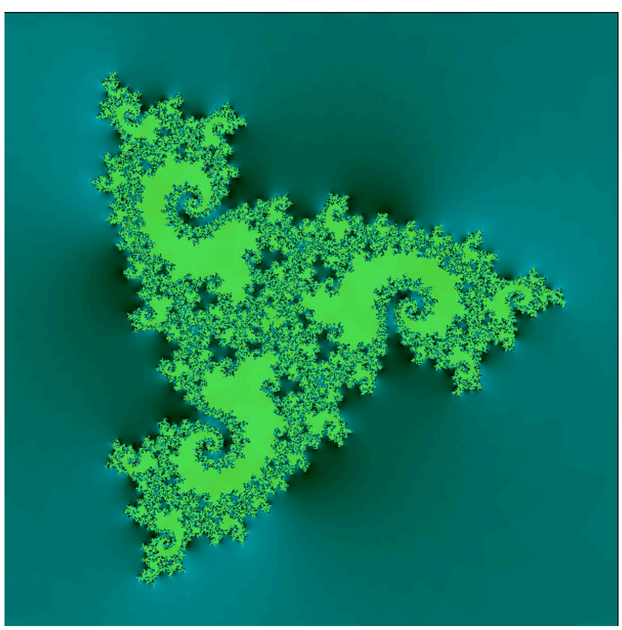

FIGURA 1. `JuliaGreen(), nrIter = 100`

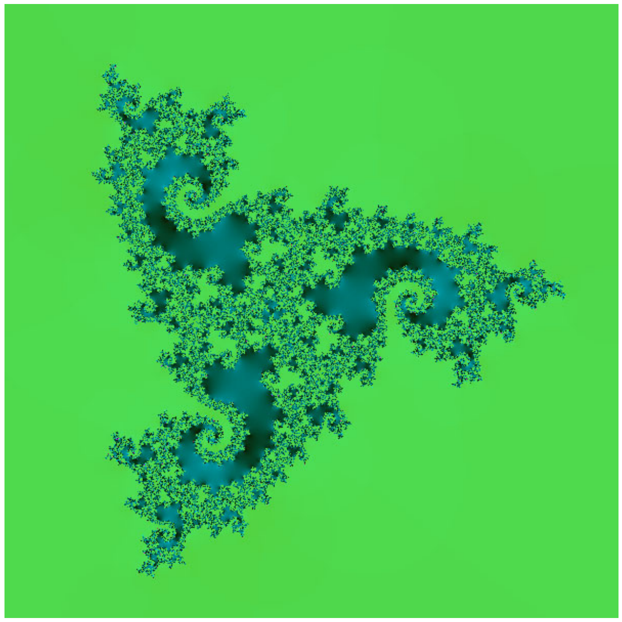

FIGURA 2. `JuliaGreen(), nrIter = 101`

a) Ce se poate deduce din faptul că cele două nuanţe de verde îşi schimbă locurile între ele?

b) Să setăm `nrIter = 1000`. Explicaţi de ce acum, în comparaţie cu cazul `nrIter = 100`, au dispărut umbrele din zona verde închis.

FIGURA 3. `JuliaGreen(), nrIter = 1000`

2. JuliaBazine(). Să considerăm funcţia

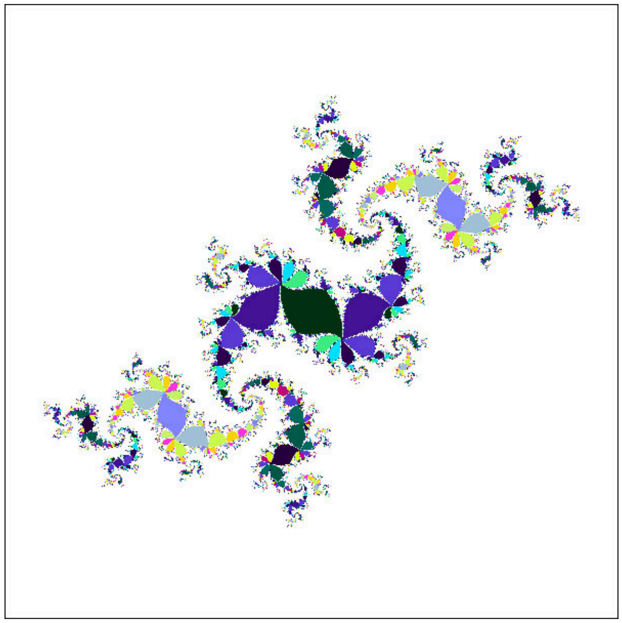

FIGURA 4. `JuliaBazine()`

```
def JuliaBazine():
    c = complex(-0.21, -0.7)

    def f(z):
        return z * z + c

    C.setXminXmaxYminYmax(-1.5, 1.5, -1.5, 1.5)
    nrIter = 1001
    rhoMax = 1.0e5
    for coloana in C.screenColumns():
        for zeta in coloana:
            z = zeta
            for k in range(nrIter):
                if C.rho(z) > rhoMax: break
                z = f(z)
            col = Color.White
            if C.rho(z) < rhoMax:
                col = Color.Index(10 * sum(C.getHK(z)) + 200)
            C.setPixel(zeta, col)
        if C.mustClose():
            return
```

care colorează prin metoda locului final punctele Fatou ale funcţiei $f_c(z) = z^2+c$, cu $c = -0.21 - 0.70i$. Sunt puse în evidenţă componentele conexe ale mulţimii Fatou, componenta albă corespunde punctului de la infinit, care în acest caz este un punct fix atractiv, iar componentele colorate corespund unei orbite periodice atractive.

Incercaţi să determinaţi, variind numărul de iteraţii `nrIter`, câte puncte distincte are această orbită periodică.

3. JuliaPlina2(). Funcţia următoare

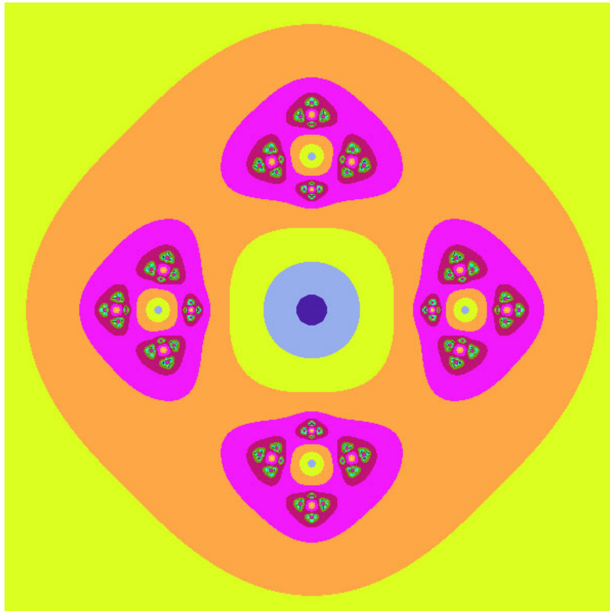

FIGURA 5. JuliaPlina2()

```
def JuliaPlina2():
    rhoMax = 1.0e2

    def f(z):
        if z == 0:
            return rhoMax
        u = z * z
        return u - 1 / u

    C.setXminXmaxYminYmax(-2, 2, -2, 2)
    nrIter = 1001
    for coloana in C.screenColumns():
        for zeta in coloana:
            z = zeta
            for k in range(nrIter):
                z = f(z)
                if C.rho(z) > rhoMax: break
            C.setPixel(zeta, Color.Index(100 * k))
        if C.mustClose():
            break
```

reprezintă grafic mulţimea Julia plină ataşată funcţiei $f(z) = z^2 - 1/z^2$. Incercaţi să explicaţi autosimilaritatea evidentă care apare în figura obţinută.

4. JuliaSierpinski(). In figura 6 este reprezentată cu negru mulţimea Ju-

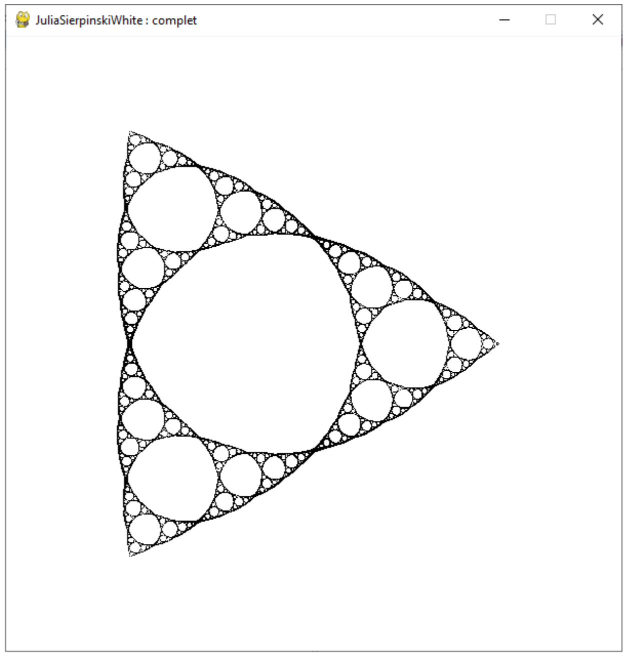

FIGURA 6. `JuliaSierpinskiWhite()`

lia ataşată funcţiei $f(z) = \frac{2(z^3-2)}{3z}$ ca frontieră a bazinului de atracţie al punctului de la infinit, bazin colorat cu alb. Incercaţi să implementaţi o funcţie `JuliaSierpinski()` care să traseze acest desen. Adăugaţi mai multe culori:

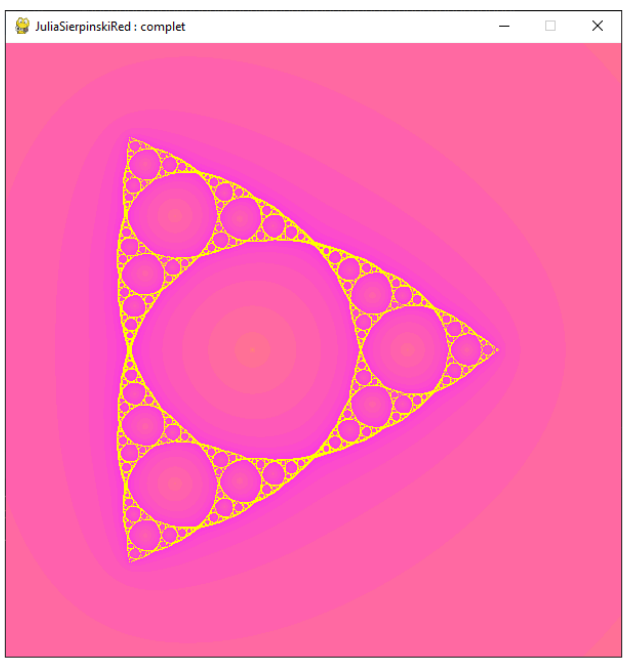

FIGURA 7. `JuliaSierpinskiRed()`

5. JuliaNautilus(). Adaptaţi funcţia `JuliaRandom()` pentru a reprezenta mulţimea Julia ataşată funcţiei

$$f(z) = \frac{z^3 - z}{\omega z^2 + 1}$$

pentru $\omega = 1.001(\cos\frac{\pi}{12} + i\sin\frac{\pi}{12})$.

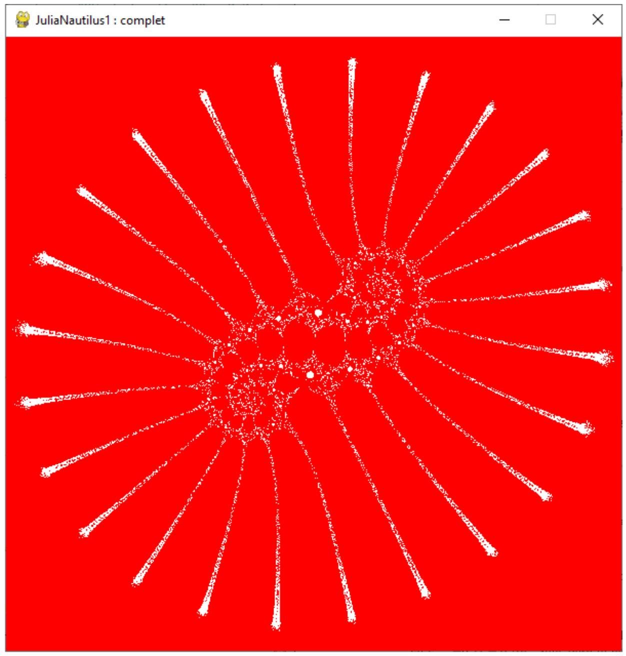

FIGURA 8. `JuliaNautilus1()`

Incercaţi apoi să analizaţi acest exemplu prin metoda locului final:

FIGURA 9. `JuliaNautilus2()`

6. Glynn(). Desenaţi mulţimea Julia plină asociată lui $f(z) = z^{1.5} + c$, pentru $c = -0.2$.

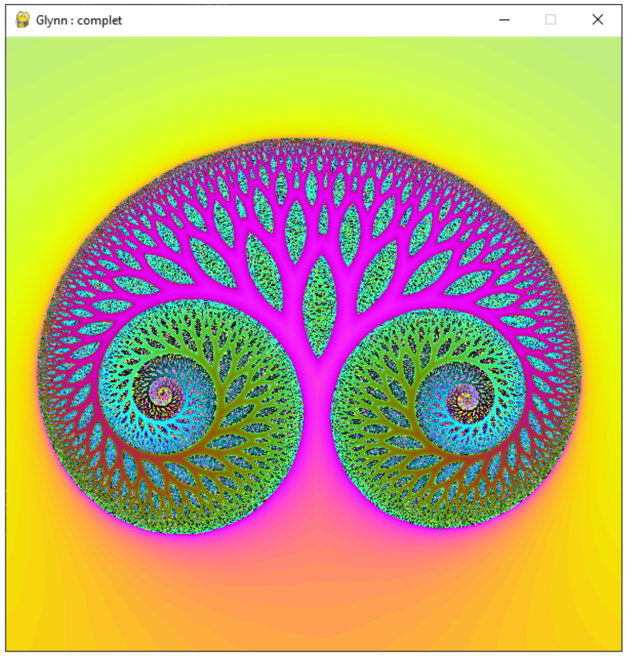

FIGURA 10. $f(z) = z^{1.5} - 0.2$

Încercaţi şi pentru $f(z) = z^{1.75} + c$, cu $c = -0.375$.

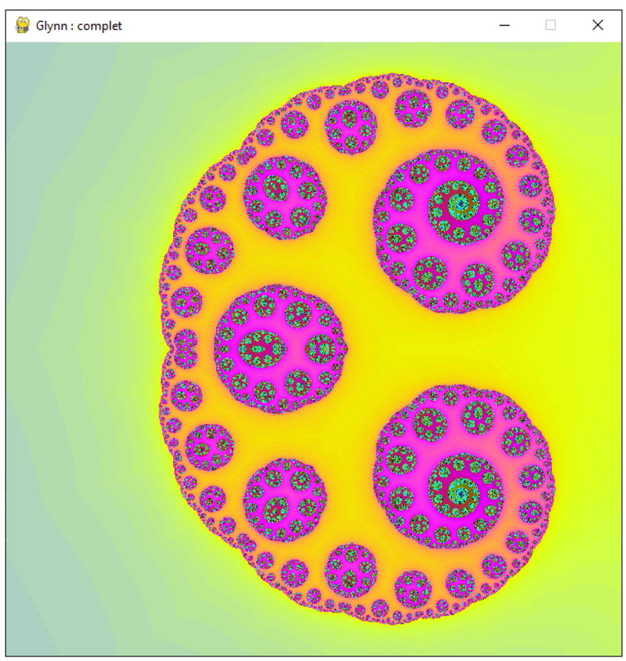

FIGURA 11. $f(z) = z^{1.75} - 0.375$

7. JuliaFinal(). Funcţia Python următoare colorează prin metoda locului final punctele Fatou ale funcţiei $f(z) = (z^3+c)/z$ pentru $c = 0.001$. Aflaţi semnificaţia celor patru culori.

```
def JuliaFinal():
    c = 0.001

    def f(z):
        if z == 0: return rhoMax
        return (z * z * z + c) / z

    C.setXminXmaxYminYmax(-1.1, 1.1, -1.1, 1.1)
    C.fillScreen(Color.Black)
    C.refreshScreen()
    rhoMax = 100
    nrIter = 107
    for coloana in C.screenColumns():
        for zeta in coloana:
            z = zeta
            for k in range(nrIter):
                if C.rho(z) > rhoMax: break
                z = f(z)
            col = Color.Red
            if C.rho(z) <= rhoMax:
                col = Color.Index(abs(sum(C.getHK(20 * z))) + 650)
            C.setPixel(zeta, col)
            C.setPixel(z, col)
        if C.mustClose():
            return
```

FIGURA 12. JuliaFinal()