Toán học tính toán: Python buổi 2/10

1 Phương pháp chia đôi

```
f = lambda x: x**3 + 2*x - 1
f(0), f(2)
```

Mã 1: Phương pháp chia đôi - Module (a)

```
a, b = 0, 2
for _ in range(5):
    c = (a + b) / 2
    if f(c) == 0:
        print(f'Nghiệm đúng: {c}')
        break
    elif f(a) * f(c) < 0:
        b = c
    else:
        a = c
    print(a, b)   # a_n, b_n với n = 1,2,...,5
                  # Để đánh giá sai số ε_n = b_n − a_n, ta in thêm b - a
```

Mã 2: Phương pháp chia đôi - Module (b, c)

```
a, b = 0, 2
while b - a > 10 ** -2:
    c = (a + b) / 2
    if f(c) == 0:
        print(f'Nghiệm đúng: {c}')
        break
    elif f(a) * f(c) < 0:
        b = c
    else:
        a = c
    print(a, b)
```

Mã 3: Phương pháp chia đôi - Module (d)

```
from sympy import *
log((2 - 0) / 10**-6, 2.)
```

Mã 4: Phương pháp chia đôi - Module (e)

2 Phương pháp Newton

```
f = lambda x: x**3 - x**2 - 3

from sympy import *
x = symbols('x')

# Cách 1
plot(f(x), (x, 1, 4))

# Cách 2
f(x).diff(x, 2)
f(x).diff().subs(x, 1)
```

Mã 5: Phương pháp Newton - Module (a)

```
t = symbols('t')
df = lambda x: f(t).diff().subs(t, x)

x = 4.
for _ in range(3):
  x = x - f(x) / df(x)
  print(x)
```

Mã 6: Phương pháp Newton - Module (b)

```
x = symbols('x')

# Cách 1
plot(abs(f(x).diff(x, 2)), (x, 1, 4))   # M = 22.5

# Cách 2
f(x).diff(x, 2)                         # M = 22
```

Mã 7: Phương pháp Newton - Đánh giá M

```
M = 22
m = min( abs(df(1)), abs(df(4)) )

x0 = 4.
for _ in range(3):
  x = x0 - f(x0) / df(x0)
  ss = M / 2 / m * (x - x0)**2  # biến x và x0 đại diện cho x_n và x_{n-1}
  x0 = x
  print(x, ss)
```

Mã 8: Phương pháp Newton - Module (c)

```
x0 = 4.
n = 0
while True:
    x = x0 - f(x0) / df(x0)
    ss = M / 2 / m * (x - x0)**2
    x0 = x
    n += 1
    print(n, x, ss)
    if ss < 10**-6:
        break
```

Mã 9: Phương pháp Newton - Module (d)

```
x = 4.
for _ in range(3):
    x = x - f(x) / df(4)
    print(x)
```

Mã 10: Phương pháp Newton - Module (e)

3 Phương pháp lặp điểm bất động

```
g = lambda x: (x**2 + 3) ** (1/3)

from sympy import *
```

```
x = symbols('x')

plot(g(x), (x, 1, 4))

plot(abs(g(x).diff()), (x, 1, 4))
q = 0.38
```

Mã 11: Phương pháp lặp điểm bất động - Module (a)

```
x = 2.5
for _ in range(3):
    x = g(x)
    print(x)
```

Mã 12: Phương pháp lặp điểm bất động - Module (b)

```
x0 = 2.5
for _ in range(3):
    x = g(x0)
    ss = q / (1-q) * abs(x - x0)
    x0 = x
    print(x, ss)
```

Mã 13: Phương pháp lặp điểm bất động - Module (c)

```
q = 0.38
x0 = 2.5
n = 0
print(n, x, ss)
while True:
    x = g(x0)
    ss = q / (1-q) * abs(x - x0)
    x0 = x
    n += 1
    print(n, x, ss)
    if ss < 10**-4:
        break
```

Mã 14: Phương pháp lặp điểm bất động - Module (d)

```
x0 = 2.5
x1 = g(x0)
log(10**-10 * (1-q) / abs(x1 - x0), q)
```

Mã 15: Phương pháp lặp điểm bất động - Module (e)