



Uporaba ChatGPT 1

Python urejevalnik se nahaja na strani:

<https://www.w3resource.com/python-exercises/python-basic-exercises.php#EDITOR>

1. Minimum funkcije na intervalu:

- Zahteva: Python program za minimum funkcije na intervalu.
- Odgovor: Tukaj je primer programa v Pythonu, ki poišče minimum funkcije na danem intervalu z uporabo metode zlatega reza:

Program

```
import math
def zlati_rez(f, a, b, tol=1e-6):
    # Izračunamo konstante alfa in beta
    alfa = (math.sqrt(5) - 1) / 2
    beta = 1 - alfa

    # Izračunamo začetne točke
    x1 = a + beta * (b - a)
    x2 = a + alfa * (b - a)
    f1, f2 = f(x1), f(x2)

    while abs(b - a) > tol:
        if f1 < f2:
            b = x2
            x2 = x1
            x1 = a + beta * (b - a)
            f2 = f1
            f1 = f(x1)
        else:
            a = x1
            x1 = x2
            x2 = a + alfa * (b - a)
            f1 = f2
            f2 = f(x2)

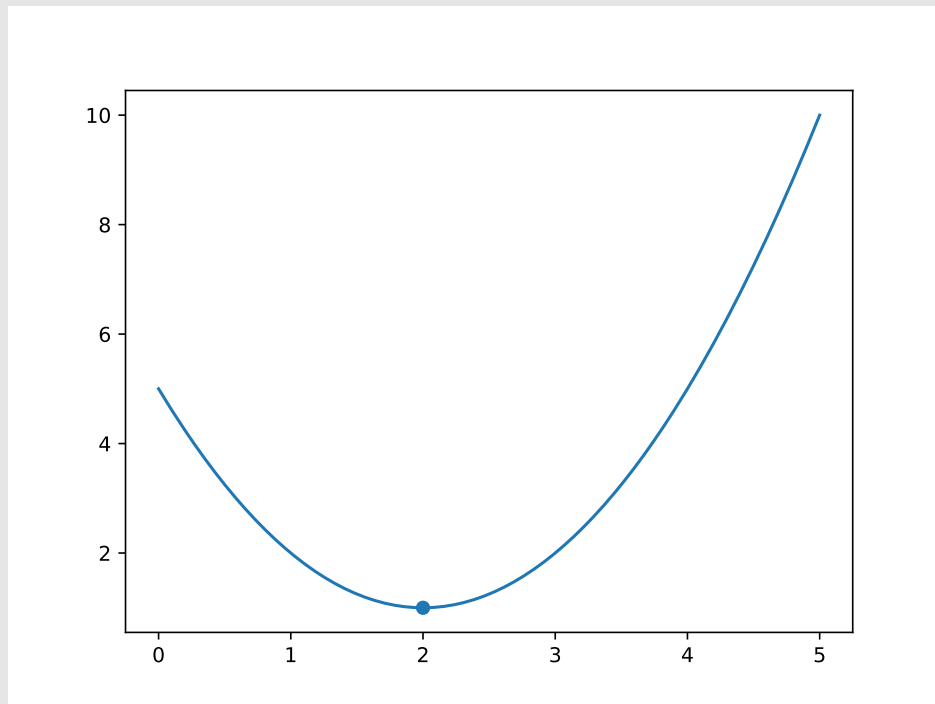
    return (a + b) / 2
```

Glavni program

```
if __name__ == '__main__':  
    # Primer uporabe  
    def funkcija(x):  
        return x**2 - 4*x + 5  
  
    a = 0  
    b = 5  
    minimum = zlati_rez(funkcija, a, b)  
    print(minimum)
```

2.000000123348785

Slika



2. Numerično reševanje enačbe

- Zahteva: python find the solution of the function $f(x)$ on the interval.
- Odgovor: To find the solution of a function in Python, you can use a numerical method such as the bisection method, Newton's method, or the secant method. Here's an example using the bisection method to find the root of a function $f(x)$ on the interval $[a, b]$:

Program

```
def bisection_method(f, a, b, tol=1e-6, max_iter=100):
    """
    Finds the root of function f(x) on the interval [a, b].
    tol: tolerance for convergence
    max_iter: maximum number of iterations
    """
    if f(a) * f(b) > 0:
        raise ValueError("f(a) and f(b) must have opposite signs")
    for i in range(max_iter):
        c = (a + b) / 2
        if abs(f(c)) < tol:
            return c
        if f(a) * f(c) < 0:
            b = c
        else:
            a = c
    raise RuntimeError("Bisection method did not converge")
```

Glavni program

```
# Example usage:
def f(x):
    return x**2 - 2
root = bisection_method(f, 0, 2)
print(root) # Output: 1.4142136573791504
```

3. Verižni ulomek racionalno število:

- Zahteva: Python program za verižni ulomek para števil (števec in imenovalc).
- Odgovor: Primer enostavnega programa, ki izračuna verižni ulomek para števil števca in imenovalca racionalnega števila.

Program

```
#!/usr/bin/env python3
def continued_fraction(a, b):
    cf = []
    if a < b:
        a, b = b, a
    cf.append(0)
    while b > 0:
        cf.append(a // b)
        # Izračun naslednjega dela ulomka
        a, b = b, a % b
    return cf
```

Glavni program

```
if __name__ == '__main__':
    a, b = 7, 5
    cf = continued_fraction(a, b)
    print(cf)

    a, b = 3141592653589793, 10000000000000000
    cf = continued_fraction(a, b)
    print(cf)
```

[1, 2, 2] [3, 7, 15, 1, 292, 1, 1, 1, 2, 1, 3, 1, 14, 4, 2, 3, 1, 12, 5, 1, 5, 20, 1, 11, 1, 1, 1, 2]

4. Verižni ulomek realno število:

- Zahteva: Python program za verižni ulomek.
- Odgovor: Primer enostavnega programa, ki izračuna verižni ulomek za dano število.

Program

```
#!/usr/bin/env python3
import math

def verizni_ulomek(x, n=10):
    a = []
    while x != int(x) and len(a) < n:
        a.append(int(x))
        n -= 1
        x = 1 / (x - int(x))
    return a
```

Glavni program

```
if __name__ == '__main__':
    # Primer uporabe
    x = 3.14159265359
    ulomek = verizni_ulomek(x)
    print(ulomek)

    x = math.sqrt(2)
    ulomek = verizni_ulomek(x, 20)
    print(ulomek)
```

[3, 7, 15, 1, 292, 1, 1, 1, 2, 1, 3, 1, 14, 4] [1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]