Problem: Give an equation $x^3 - 5x + 3 = 0$:

1.Determine the two positive roots to 4 decimal places by using the bisection method.

2.Determine the two roots to more accurate (14) decimal places by using the Newton-Raphson method.

3.Determine the two positive roots to 14 decimal places by using the hybrid method.

We firstly define the function of bisection method. The function needs 2 inputs, the first x_1 refers to the lower bound and the second x_2 refers to the upper bound. In this question, we need to select and input the x_1, x_2 by ourselves. And roughly we can estimate that the first root is between $[0,1]$ and the second root is between $[1,2]$. Inside the function, we define $x_3 = \frac{x_1+x_2}{2}$. And then, we use a while-loop, the while-loop continues when $f(x_3) > 1 \times 10^{-4}$. Inside the while-loop, we use an if-else structure: if $f(x_1) * f(x_3) < 0$ (which means the root is in $[x_1, x_3]$), then $x_2 \leftarrow x_3, x_3 \leftarrow \frac{x_1+x_2}{2}$; else (which means the root is in $[x_3, x_2]$) $x_1 \leftarrow x_3, x_3 \leftarrow \frac{x_1+x_2}{2}$. This function will finally return a rough solution x of the equation, which satisfies $f(x) < 1 \times 10^{-4}$.

Then we define the function of Newton method. The function takes 1 parameter t as input, which is the original value of the iteration. In this question, we can use the rough solution returned from the bisection method as the input. Inside the function, we use a while-loop. We define $a = f(t), b = f'(t)$. If $|a| > 1 \times 10^{-14}$, $x \leftarrow x - \frac{a}{b}$; else (which means $|a| \leq 1 \times 10^{-14}$) return x and break the while-loop. In this way, we get an accurate solution of the equation.

Finally, we define the function of hybrid method. The function needs 3 inputs: the first x_1 refers to the lower bound, the second x_2 refers to the upper bound, and the third t refers to the original value we set for Newton method. In program, we set 3 parameters in advance. Roughly, we can estimate that the first root is between $[0,1]$ and the second root is between $[1,2]$. So, for the first time we use hybrid method, we set $x_1 = 0, x_2 = 1, t = \frac{x_1+x_2}{2} = 0.5$. And for the second time we use hybrid method, we set $x_1 = 1, x_2 = 2, t = \frac{x_1+x_2}{2} = 1.5$.

The basic structure of hybrid method is similar to bisection method and Newton method, and we also use while-loop inside the function. However, in each loop, before we use Newton method and set $x \leftarrow x - \frac{f(t)}{f'(t)}$, we firstly check whether the new value $x - \frac{f(t)}{f'(t)}$ sits in the interval which is given by bisection method. If it sits in the interval $[x_1, x_2]$, then $x \leftarrow x - \frac{f(t)}{f'(t)}$. If not, $x \leftarrow \frac{x_1+x_2}{2}$.

Pseudo-code of Bisection Method & Newton Method

Input: $f(x) = x^3 - 5x + 3$, lower bound x_1, upper bound x_2

Output: The solution x of the equation $f(x) = x^3 - 5x + 3 = 0$

1. $x_3 \leftarrow \frac{x_1+x_2}{2}$
2. **While** $f(x_3) > 1 \times 10^{-4}$ **Do**
3. **If** $f(x_1) * f(x_3) < 0$ **Then**
4. $x_2 \leftarrow x_3, x_3 \leftarrow \frac{x_1+x_2}{2}$
5. **Else**
6. $x_1 \leftarrow x_3, x_3 \leftarrow \frac{x_1+x_2}{2}$
7. **Return** x_3
8. $x \leftarrow x_3$
9. **While** True **Do**
10. $a \leftarrow f(x)$
11. $b \leftarrow f'(x)$
12. **If** $|a| > 1 \times 10^{-14}$ **Then**
13. $x \leftarrow x - \frac{a}{b}$
14. **Else**
15. **Return** x
16. **Break**

Pseudo-code of Hybrid Method

Input: $f(x) = x^3 - 5x + 3$, lower bound x_1, upper bound x_2, original value t

Output: The solution sol of the equation $f(x) = x^3 - 5x + 3 = 0$

$x_3 \leftarrow \frac{x_1+x_2}{2}, a \leftarrow f(t), b \leftarrow f'(t)$

While $|a| > 1 \times 10^{-14}$ and $f(x_3) > 1 \times 10^{-14}$ **Do**

If $f(x_1) * f(x_3) < 0$ **Then**

$x_2 \leftarrow x_3,\ x_3 \leftarrow \frac{x_1+x_2}{2}$

If $x_1 < t - \frac{a}{b} < x_2$ **Then**

$t \leftarrow t - \frac{a}{b}$

Else

$t \leftarrow x_3$

Else

$x_1 \leftarrow x_3,\ x_3 \leftarrow \frac{x_1+x_2}{2}$

If $x_1 < t - \frac{a}{b} < x_2$ **Then**

$t \leftarrow t - \frac{a}{b}$

Else

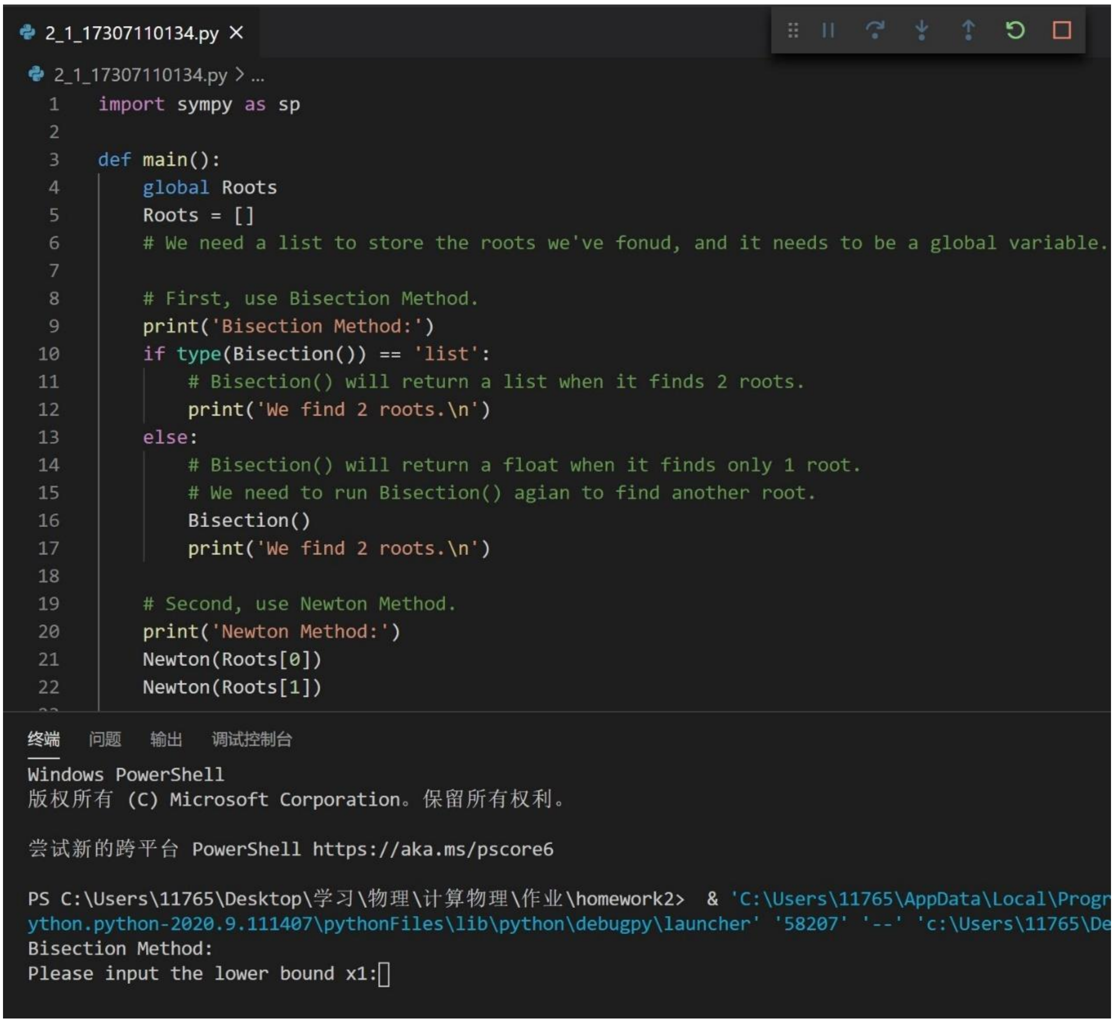
$t \leftarrow x_3$

$a \leftarrow f(t), b \leftarrow f'(t)$

/*In the end of the loop, we need to update the value of *a* and *b**/

$sol \leftarrow t$

Return sol



As we've discussed before, for the first root, we input $x_1 = 0, x_2 = 1$. And for the second root, we input $x_1 = 1, x_2 = 2$. If you input x_1, x_2 which makes $f(x_1) * f(x_2) > 0$, the program will need you to input again.

The picture showed below is the result of the program.

2_1_17307110134.py ×

2_1_17307110134.py > ...

```python
import sympy as sp

def main():
    global Roots
    Roots = []
    # We need a list to store the roots we've fonud, and it needs to be a global variable.

    # First, use Bisection Method.
    print('Bisection Method:')
    if type(Bisection()) == 'list':
        # Bisection() will return a list when it finds 2 roots.
        print('We find 2 roots.\n')
    else:
        # Bisection() will return a float when it finds only 1 root.
        # We need to run Bisection() agian to find another root.
        Bisection()
```

终端　问题　输出　调试控制台

```
PS C:\Users\11765\Desktop\学习\物理\计算物理\作业\homework2>  & 'C:\Users\11765\AppData\Local\Progra
ython.python-2020.9.111407\pythonFiles\lib\python\debugpy\launcher' '57636' '--' 'c:\Users\11765\Des
Bisection Method:
Please input the lower bound x1:0
Please input the upper bound x2:1
0.6566 is a root of the equation.
Please input the lower bound x1:1
Please input the upper bound x2:2
1.8342 is a root of the equation.
We find 2 roots.

Newton Method:
f(x)=0.00001564207741, f'(x)=-3.70656545460224, the times of iteration is 0.
f(x)=0.00000000003508, f'(x)=-3.70654882863164, the times of iteration is 1.
f(x)=0.00000000000000, f'(x)=-3.70654882860254, the times of iteration is 2.
0.65662043104711 is a root of the equation.

f(x)=-0.00007471149729, f'(x)=5.09318274259567, the times of iteration is 0.
f(x)=0.00000000118406, f'(x)=5.09334418014623, the times of iteration is 1.
f(x)=-0.00000000000000, f'(x)=5.09334417758510, the times of iteration is 2.
1.83424318431392 is a root of the equation.

Hybrid Method:
0.65662043104711 is a root of the equation.
1.83424318431392 is a root of the equation.
```