

Fibonacci search method screen shot:

```
After 1, value of function is -6.9858276643998935
new ranges of inputs are [[2.5, 3.119847619847619], [1.5, 2.119847619847619]]
After 2, value of function is -6.9858276643998935
new ranges of inputs are [[2.738895238895238, 3.119847619847619], [1.7388952388952381, 2.119847619847619]]
After 3, value of function is -6.999433186575964
new ranges of inputs are [[2.888952388952381, 3.119847619847619], [1.888952388952381, 2.119847619847619]]
After 4, value of function is -6.999433186575964
new ranges of inputs are [[2.9761984761984763, 3.119847619847619], [1.9761984761984763, 2.119847619847619]]
After 5, value of function is -6.999433186575964
new ranges of inputs are [[2.9761984761984763, 3.0714285714285716], [1.9761984761984763, 2.0714285714285716]]

Process finished with exit code 0
```

Golden section search method screen shot:

```
After 1, value of function is -6.98606797749979
new ranges of inputs are [[2.881966011250105, 3.5], [1.881966011250105, 2.5]]
After 2, value of function is -6.98606797749979
new ranges of inputs are [[2.881966011250105, 3.26393202250021], [1.881966011250105, 2.26393202250021]]
After 3, value of function is -6.999223594996215
new ranges of inputs are [[2.881966011250105, 3.118033988749895], [1.881966011250105, 2.118033988749895]]
After 4, value of function is -6.999223594996215
new ranges of inputs are [[2.9721359549995796, 3.118033988749895], [1.9721359549995794, 2.118033988749895]]
After 5, value of function is -6.999223594996215
new ranges of inputs are [[2.9721359549995796, 3.0623058987490537], [1.9721359549995794, 2.0623058987490537]]
After 6, value of function is -6.9999567324320715
new ranges of inputs are [[2.9721359549995796, 3.027864045000421], [1.9721359549995794, 2.027864045000421]]
After 7, value of function is -6.999956732432072
new ranges of inputs are [[2.9721359549995796, 3.0065778087482125], [1.9721359549995794, 2.0065778087482125]]
After 8, value of function is -6.9999567324320715
new ranges of inputs are [[2.9852915724960045, 3.0065778087482125], [1.9852915724960043, 2.0065778087482125]]
After 9, value of function is -6.99999758878108
new ranges of inputs are [[2.993422191251788, 3.0065778087482125], [1.9934221912517875, 2.0065778087482125]]
After 10, value of function is -6.999997588781081
new ranges of inputs are [[2.993422191251788, 3.001552810007571], [1.9934221912517875, 2.001552810007571]]

Process finished with exit code 0
```

Nelder-Mead method screen shot:

```
After i = 1, minimum value is:-3.36 and input vectors are [array([1.2, 0. ]), array([0. , 0.8]), array([0. , 0.])]
After i = 2, minimum value is:-5.88 and input vectors are [array([1.8, 1.2]), array([1.2, 0. ]), array([0. , 0.8])]
After i = 3, minimum value is:-5.88 and input vectors are [array([1.8, 1.2]), array([3. , 0.4]), array([1.2, 0. ])]
After i = 4, minimum value is:-6.240000000000001 and input vectors are [array([3.6, 1.6]), array([1.8, 1.2]), array([3. , 0.4])]
After i = 5, minimum value is:-6.240000000000001 and input vectors are [array([3.6, 1.6]), array([2.4, 2.4]), array([1.8, 1.2])]
After i = 6, minimum value is:-6.72 and input vectors are [array([2.4, 1.6]), array([3.6, 1.6]), array([2.4, 2.4])]
After i = 7, minimum value is:-6.9099999999999998 and input vectors are [array([2.7, 2. ]), array([2.4, 1.6]), array([3.6, 1.6])]
After i = 8, minimum value is:-6.9099999999999998 and input vectors are [array([2.7, 2. ]), array([3.075, 1.7 ]), array([2.4, 1.6])]
After i = 9, minimum value is:-6.9099999999999998 and input vectors are [array([2.7, 2. ]), array([3.375, 2.1 ]), array([3.075, 1.7 ])]
After i = 10, minimum value is:-6.9741796874999995 and input vectors are [array([3.05625, 1.875 ]), array([2.7, 2. ]), array([3.375, 2.1 ])]
Terminating the iteration with the following vectors
[3.05625 1.875 ]
[2.7 2. ]
[3.1265625 2.01875 ]
Smallest value is -6.9741796874999995 with vector [3.05625 1.875 ]

Process finished with exit code 0
```