Вариант: 2-3-1

1. gcd(87, 30) = 3

1 def gcd(x=87, y=30)

2 if 87 < 0: --- False

4 if 30 < 0: --- False

6 if 87 == 0: --- False

8 while 30 != 0: --- True

9 rem = 87 % 30

rem = 27

10 x = 30

11 y = 27

8 while 27 != 0: --- True

9 rem = 30 % 27

rem = 3

10 x = 27

11 y = 3

8 while 3 != 0: --- True

9 rem = 27 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(0, 0) = 0

1 def gcd(x=0, y=0)

2 if 0 < 0: --- False

4 if 0 < 0: --- False

6 if 0 == 0: --- True

7 return 0

3. hex(192) = 'C0'

3 def hex(number=192)

4 if 192 == 0: --- False

6 res = ''

7 while 192 > 0: --- True

8 digit = 192 % 16

digit = 0

9 res = DIGITS[0] + ''

res = '0'

10 number = 192 // 16

number = 12

7 while 12 > 0: --- True

8 digit = 12 % 16

digit = 12

9 res = DIGITS[12] + '0'

res = 'C0'

10 number = 12 // 16

number = 0

7 while 0 > 0: --- False

11 return 'C0'

4. square\_equal(-40, -46, -13) = [-0.5, -0.65]

3 def square\_equal(a=-40, b=-46, c=-13)

4 if -40 != 0: --- True

5 D = -46\*-46 - 4\*-40\*-13

D = 36

6 if 36 > 0: --- True

7 x1 = (--46 - sqrt(36)) / (2\*-40)

x1 = -0.5

8 x2 = (--46 + sqrt(36)) / (2\*-40)

x2 = -0.65

9 return [-0.5, -0.65]

5. square\_equal(-72, 37, -67) = []

3 def square\_equal(a=-72, b=37, c=-67)

4 if -72 != 0: --- True

5 D = 37\*37 - 4\*-72\*-67

D = -17927

6 if -17927 > 0: --- False

10 elif -17927 == 0: --- False

12 else:

13 return []

6. findmax([95, -23, 5, 90, -31]) = 95

1 def findmax(items=[95, -23, 5, 90, -31])

2 if len([95, -23, 5, 90, -31]) == 0: --- False

4 m = items[0]

m = 95

5 i = 1

6 while 1 < len([95, -23, 5, 90, -31]): --- True

7 if 95 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([95, -23, 5, 90, -31]): --- True

7 if 95 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([95, -23, 5, 90, -31]): --- True

7 if 95 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([95, -23, 5, 90, -31]): --- True

7 if 95 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([95, -23, 5, 90, -31]): --- False

10 return 95

7. unique([2, 2, 1, -31]) = [2, 1, -31]

1 def unique(items=[2, 2, 1, -31])

2 res = []

3 i = 0

4 while 0 < len([2, 2, 1, -31]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [2]

7 i = 0 + 1

i = 1

4 while 1 < len([2, 2, 1, -31]): --- True

5 if len([2]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([2, 2, 1, -31]): --- True

5 if len([2]) == 0 or res[-1] != items[2]: --- True

6 res = [2] + [items[2]]

res = [2, 1]

7 i = 2 + 1

i = 3

4 while 3 < len([2, 2, 1, -31]): --- True

5 if len([2, 1]) == 0 or res[-1] != items[3]: --- True

6 res = [2, 1] + [items[3]]

res = [2, 1, -31]

7 i = 3 + 1

i = 4

4 while 4 < len([2, 2, 1, -31]): --- False

8 return [2, 1, -31]

8. join(';', [31, 52, 23, 54]) = '31;52;23;54'

1 def join(sep=;, items=[31, 52, 23, 54])

2 res = ''

3 if len([31, 52, 23, 54]) > 0: --- True

4 res = str(items[0])

res = '31'

5 items = items[1:]

items = [52, 23, 54]

6 while len([52, 23, 54]) > 0: --- True

7 res = '31' + ';' + str(items[0])

res = '31;52'

8 items = items[1:]

items = [23, 54]

6 while len([23, 54]) > 0: --- True

7 res = '31;52' + ';' + str(items[0])

res = '31;52;23'

8 items = items[1:]

items = [54]

6 while len([54]) > 0: --- True

7 res = '31;52;23' + ';' + str(items[0])

res = '31;52;23;54'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '31;52;23;54'

Вариант: 2-3-2

1. gcd(-84, -70) = 14

1 def gcd(x=-84, y=-70)

2 if -84 < 0: --- True

3 x = --84

x = 84

4 if -70 < 0: --- True

5 y = --70

y = 70

6 if 84 == 0: --- False

8 while 70 != 0: --- True

9 rem = 84 % 70

rem = 14

10 x = 70

11 y = 14

8 while 14 != 0: --- True

9 rem = 70 % 14

rem = 0

10 x = 14

11 y = 0

8 while 0 != 0: --- False

12 return 14

2. gcd(0, -76) = 76

1 def gcd(x=0, y=-76)

2 if 0 < 0: --- False

4 if -76 < 0: --- True

5 y = --76

y = 76

6 if 0 == 0: --- True

7 return 76

3. hex(253) = 'FD'

3 def hex(number=253)

4 if 253 == 0: --- False

6 res = ''

7 while 253 > 0: --- True

8 digit = 253 % 16

digit = 13

9 res = DIGITS[13] + ''

res = 'D'

10 number = 253 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + 'D'

res = 'FD'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'FD'

4. square\_equal(4, 7, -92) = [-5.75, 4.0]

3 def square\_equal(a=4, b=7, c=-92)

4 if 4 != 0: --- True

5 D = 7\*7 - 4\*4\*-92

D = 1521

6 if 1521 > 0: --- True

7 x1 = (-7 - sqrt(1521)) / (2\*4)

x1 = -5.75

8 x2 = (-7 + sqrt(1521)) / (2\*4)

x2 = 4.0

9 return [-5.75, 4.0]

5. square\_equal(-23, 6, -31) = []

3 def square\_equal(a=-23, b=6, c=-31)

4 if -23 != 0: --- True

5 D = 6\*6 - 4\*-23\*-31

D = -2816

6 if -2816 > 0: --- False

10 elif -2816 == 0: --- False

12 else:

13 return []

6. findmax([-26, -11, -2, -67, -3]) = -2

1 def findmax(items=[-26, -11, -2, -67, -3])

2 if len([-26, -11, -2, -67, -3]) == 0: --- False

4 m = items[0]

m = -26

5 i = 1

6 while 1 < len([-26, -11, -2, -67, -3]): --- True

7 if -26 < items[1]: --- True

8 m = items[1]

m = -11

9 i = 1 + 1

i = 2

6 while 2 < len([-26, -11, -2, -67, -3]): --- True

7 if -11 < items[2]: --- True

8 m = items[2]

m = -2

9 i = 2 + 1

i = 3

6 while 3 < len([-26, -11, -2, -67, -3]): --- True

7 if -2 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-26, -11, -2, -67, -3]): --- True

7 if -2 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-26, -11, -2, -67, -3]): --- False

10 return -2

7. unique([-61, 76, 76]) = [-61, 76]

1 def unique(items=[-61, 76, 76])

2 res = []

3 i = 0

4 while 0 < len([-61, 76, 76]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [-61]

7 i = 0 + 1

i = 1

4 while 1 < len([-61, 76, 76]): --- True

5 if len([-61]) == 0 or res[-1] != items[1]: --- True

6 res = [-61] + [items[1]]

res = [-61, 76]

7 i = 1 + 1

i = 2

4 while 2 < len([-61, 76, 76]): --- True

5 if len([-61, 76]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-61, 76, 76]): --- False

8 return [-61, 76]

8. join(',', [15, 96, 94, 30]) = '15,96,94,30'

1 def join(sep=,, items=[15, 96, 94, 30])

2 res = ''

3 if len([15, 96, 94, 30]) > 0: --- True

4 res = str(items[0])

res = '15'

5 items = items[1:]

items = [96, 94, 30]

6 while len([96, 94, 30]) > 0: --- True

7 res = '15' + ',' + str(items[0])

res = '15,96'

8 items = items[1:]

items = [94, 30]

6 while len([94, 30]) > 0: --- True

7 res = '15,96' + ',' + str(items[0])

res = '15,96,94'

8 items = items[1:]

items = [30]

6 while len([30]) > 0: --- True

7 res = '15,96,94' + ',' + str(items[0])

res = '15,96,94,30'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '15,96,94,30'

Вариант: 2-3-3

1. gcd(36, 69) = 3

1 def gcd(x=36, y=69)

2 if 36 < 0: --- False

4 if 69 < 0: --- False

6 if 36 == 0: --- False

8 while 69 != 0: --- True

9 rem = 36 % 69

rem = 36

10 x = 69

11 y = 36

8 while 36 != 0: --- True

9 rem = 69 % 36

rem = 33

10 x = 36

11 y = 33

8 while 33 != 0: --- True

9 rem = 36 % 33

rem = 3

10 x = 33

11 y = 3

8 while 3 != 0: --- True

9 rem = 33 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(0, 86) = 86

1 def gcd(x=0, y=86)

2 if 0 < 0: --- False

4 if 86 < 0: --- False

6 if 0 == 0: --- True

7 return 86

3. hex(194) = 'C2'

3 def hex(number=194)

4 if 194 == 0: --- False

6 res = ''

7 while 194 > 0: --- True

8 digit = 194 % 16

digit = 2

9 res = DIGITS[2] + ''

res = '2'

10 number = 194 // 16

number = 12

7 while 12 > 0: --- True

8 digit = 12 % 16

digit = 12

9 res = DIGITS[12] + '2'

res = 'C2'

10 number = 12 // 16

number = 0

7 while 0 > 0: --- False

11 return 'C2'

4. square\_equal(0, -4, 90) = [22.5]

3 def square\_equal(a=0, b=-4, c=90)

4 if 0 != 0: --- False

14 else:

15 if -4 != 0: --- True

16 return [22.5]

5. square\_equal(-25, 78, -81) = []

3 def square\_equal(a=-25, b=78, c=-81)

4 if -25 != 0: --- True

5 D = 78\*78 - 4\*-25\*-81

D = -2016

6 if -2016 > 0: --- False

10 elif -2016 == 0: --- False

12 else:

13 return []

6. findmax([-30, -26, 14, 36]) = 36

1 def findmax(items=[-30, -26, 14, 36])

2 if len([-30, -26, 14, 36]) == 0: --- False

4 m = items[0]

m = -30

5 i = 1

6 while 1 < len([-30, -26, 14, 36]): --- True

7 if -30 < items[1]: --- True

8 m = items[1]

m = -26

9 i = 1 + 1

i = 2

6 while 2 < len([-30, -26, 14, 36]): --- True

7 if -26 < items[2]: --- True

8 m = items[2]

m = 14

9 i = 2 + 1

i = 3

6 while 3 < len([-30, -26, 14, 36]): --- True

7 if 14 < items[3]: --- True

8 m = items[3]

m = 36

9 i = 3 + 1

i = 4

6 while 4 < len([-30, -26, 14, 36]): --- False

10 return 36

7. unique([5, 5, 41, 5]) = [5, 41, 5]

1 def unique(items=[5, 5, 41, 5])

2 res = []

3 i = 0

4 while 0 < len([5, 5, 41, 5]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [5]

7 i = 0 + 1

i = 1

4 while 1 < len([5, 5, 41, 5]): --- True

5 if len([5]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([5, 5, 41, 5]): --- True

5 if len([5]) == 0 or res[-1] != items[2]: --- True

6 res = [5] + [items[2]]

res = [5, 41]

7 i = 2 + 1

i = 3

4 while 3 < len([5, 5, 41, 5]): --- True

5 if len([5, 41]) == 0 or res[-1] != items[3]: --- True

6 res = [5, 41] + [items[3]]

res = [5, 41, 5]

7 i = 3 + 1

i = 4

4 while 4 < len([5, 5, 41, 5]): --- False

8 return [5, 41, 5]

8. join('+', [35, 3, 81]) = '35+3+81'

1 def join(sep=+, items=[35, 3, 81])

2 res = ''

3 if len([35, 3, 81]) > 0: --- True

4 res = str(items[0])

res = '35'

5 items = items[1:]

items = [3, 81]

6 while len([3, 81]) > 0: --- True

7 res = '35' + '+' + str(items[0])

res = '35+3'

8 items = items[1:]

items = [81]

6 while len([81]) > 0: --- True

7 res = '35+3' + '+' + str(items[0])

res = '35+3+81'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '35+3+81'

Вариант: 2-3-4

1. gcd(-12, 40) = 4

1 def gcd(x=-12, y=40)

2 if -12 < 0: --- True

3 x = --12

x = 12

4 if 40 < 0: --- False

6 if 12 == 0: --- False

8 while 40 != 0: --- True

9 rem = 12 % 40

rem = 12

10 x = 40

11 y = 12

8 while 12 != 0: --- True

9 rem = 40 % 12

rem = 4

10 x = 12

11 y = 4

8 while 4 != 0: --- True

9 rem = 12 % 4

rem = 0

10 x = 4

11 y = 0

8 while 0 != 0: --- False

12 return 4

2. gcd(0, -77) = 77

1 def gcd(x=0, y=-77)

2 if 0 < 0: --- False

4 if -77 < 0: --- True

5 y = --77

y = 77

6 if 0 == 0: --- True

7 return 77

3. hex(227) = 'E3'

3 def hex(number=227)

4 if 227 == 0: --- False

6 res = ''

7 while 227 > 0: --- True

8 digit = 227 % 16

digit = 3

9 res = DIGITS[3] + ''

res = '3'

10 number = 227 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + '3'

res = 'E3'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'E3'

4. square\_equal(100, -4, 0) = [0.0, 0.04]

3 def square\_equal(a=100, b=-4, c=0)

4 if 100 != 0: --- True

5 D = -4\*-4 - 4\*100\*0

D = 16

6 if 16 > 0: --- True

7 x1 = (--4 - sqrt(16)) / (2\*100)

x1 = 0.0

8 x2 = (--4 + sqrt(16)) / (2\*100)

x2 = 0.04

9 return [0.0, 0.04]

5. square\_equal(-85, -63, -100) = []

3 def square\_equal(a=-85, b=-63, c=-100)

4 if -85 != 0: --- True

5 D = -63\*-63 - 4\*-85\*-100

D = -30031

6 if -30031 > 0: --- False

10 elif -30031 == 0: --- False

12 else:

13 return []

6. findmax([-98, -54, -34, 10, -35]) = 10

1 def findmax(items=[-98, -54, -34, 10, -35])

2 if len([-98, -54, -34, 10, -35]) == 0: --- False

4 m = items[0]

m = -98

5 i = 1

6 while 1 < len([-98, -54, -34, 10, -35]): --- True

7 if -98 < items[1]: --- True

8 m = items[1]

m = -54

9 i = 1 + 1

i = 2

6 while 2 < len([-98, -54, -34, 10, -35]): --- True

7 if -54 < items[2]: --- True

8 m = items[2]

m = -34

9 i = 2 + 1

i = 3

6 while 3 < len([-98, -54, -34, 10, -35]): --- True

7 if -34 < items[3]: --- True

8 m = items[3]

m = 10

9 i = 3 + 1

i = 4

6 while 4 < len([-98, -54, -34, 10, -35]): --- True

7 if 10 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-98, -54, -34, 10, -35]): --- False

10 return 10

7. unique([68, 68, 50]) = [68, 50]

1 def unique(items=[68, 68, 50])

2 res = []

3 i = 0

4 while 0 < len([68, 68, 50]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [68]

7 i = 0 + 1

i = 1

4 while 1 < len([68, 68, 50]): --- True

5 if len([68]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([68, 68, 50]): --- True

5 if len([68]) == 0 or res[-1] != items[2]: --- True

6 res = [68] + [items[2]]

res = [68, 50]

7 i = 2 + 1

i = 3

4 while 3 < len([68, 68, 50]): --- False

8 return [68, 50]

8. join(';', [48, 85, 72]) = '48;85;72'

1 def join(sep=;, items=[48, 85, 72])

2 res = ''

3 if len([48, 85, 72]) > 0: --- True

4 res = str(items[0])

res = '48'

5 items = items[1:]

items = [85, 72]

6 while len([85, 72]) > 0: --- True

7 res = '48' + ';' + str(items[0])

res = '48;85'

8 items = items[1:]

items = [72]

6 while len([72]) > 0: --- True

7 res = '48;85' + ';' + str(items[0])

res = '48;85;72'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '48;85;72'

Вариант: 2-3-5

1. gcd(18, -60) = 6

1 def gcd(x=18, y=-60)

2 if 18 < 0: --- False

4 if -60 < 0: --- True

5 y = --60

y = 60

6 if 18 == 0: --- False

8 while 60 != 0: --- True

9 rem = 18 % 60

rem = 18

10 x = 60

11 y = 18

8 while 18 != 0: --- True

9 rem = 60 % 18

rem = 6

10 x = 18

11 y = 6

8 while 6 != 0: --- True

9 rem = 18 % 6

rem = 0

10 x = 6

11 y = 0

8 while 0 != 0: --- False

12 return 6

2. gcd(72, 0) = 72

1 def gcd(x=72, y=0)

2 if 72 < 0: --- False

4 if 0 < 0: --- False

6 if 72 == 0: --- False

8 while 0 != 0: --- False

12 return 72

3. hex(200) = 'C8'

3 def hex(number=200)

4 if 200 == 0: --- False

6 res = ''

7 while 200 > 0: --- True

8 digit = 200 % 16

digit = 8

9 res = DIGITS[8] + ''

res = '8'

10 number = 200 // 16

number = 12

7 while 12 > 0: --- True

8 digit = 12 % 16

digit = 12

9 res = DIGITS[12] + '8'

res = 'C8'

10 number = 12 // 16

number = 0

7 while 0 > 0: --- False

11 return 'C8'

4. square\_equal(2, -10, -12) = [-1.0, 6.0]

3 def square\_equal(a=2, b=-10, c=-12)

4 if 2 != 0: --- True

5 D = -10\*-10 - 4\*2\*-12

D = 196

6 if 196 > 0: --- True

7 x1 = (--10 - sqrt(196)) / (2\*2)

x1 = -1.0

8 x2 = (--10 + sqrt(196)) / (2\*2)

x2 = 6.0

9 return [-1.0, 6.0]

5. square\_equal(-75, 64, -92) = []

3 def square\_equal(a=-75, b=64, c=-92)

4 if -75 != 0: --- True

5 D = 64\*64 - 4\*-75\*-92

D = -23504

6 if -23504 > 0: --- False

10 elif -23504 == 0: --- False

12 else:

13 return []

6. findmax([-65, 4, -2, -31]) = 4

1 def findmax(items=[-65, 4, -2, -31])

2 if len([-65, 4, -2, -31]) == 0: --- False

4 m = items[0]

m = -65

5 i = 1

6 while 1 < len([-65, 4, -2, -31]): --- True

7 if -65 < items[1]: --- True

8 m = items[1]

m = 4

9 i = 1 + 1

i = 2

6 while 2 < len([-65, 4, -2, -31]): --- True

7 if 4 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-65, 4, -2, -31]): --- True

7 if 4 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-65, 4, -2, -31]): --- False

10 return 4

7. unique([-92, -92, -97]) = [-92, -97]

1 def unique(items=[-92, -92, -97])

2 res = []

3 i = 0

4 while 0 < len([-92, -92, -97]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [-92]

7 i = 0 + 1

i = 1

4 while 1 < len([-92, -92, -97]): --- True

5 if len([-92]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-92, -92, -97]): --- True

5 if len([-92]) == 0 or res[-1] != items[2]: --- True

6 res = [-92] + [items[2]]

res = [-92, -97]

7 i = 2 + 1

i = 3

4 while 3 < len([-92, -92, -97]): --- False

8 return [-92, -97]

8. join(',', [14, 53, 7]) = '14,53,7'

1 def join(sep=,, items=[14, 53, 7])

2 res = ''

3 if len([14, 53, 7]) > 0: --- True

4 res = str(items[0])

res = '14'

5 items = items[1:]

items = [53, 7]

6 while len([53, 7]) > 0: --- True

7 res = '14' + ',' + str(items[0])

res = '14,53'

8 items = items[1:]

items = [7]

6 while len([7]) > 0: --- True

7 res = '14,53' + ',' + str(items[0])

res = '14,53,7'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '14,53,7'

Вариант: 2-3-6

1. gcd(-69, 21) = 3

1 def gcd(x=-69, y=21)

2 if -69 < 0: --- True

3 x = --69

x = 69

4 if 21 < 0: --- False

6 if 69 == 0: --- False

8 while 21 != 0: --- True

9 rem = 69 % 21

rem = 6

10 x = 21

11 y = 6

8 while 6 != 0: --- True

9 rem = 21 % 6

rem = 3

10 x = 6

11 y = 3

8 while 3 != 0: --- True

9 rem = 6 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(100, 0) = 100

1 def gcd(x=100, y=0)

2 if 100 < 0: --- False

4 if 0 < 0: --- False

6 if 100 == 0: --- False

8 while 0 != 0: --- False

12 return 100

3. hex(176) = 'B0'

3 def hex(number=176)

4 if 176 == 0: --- False

6 res = ''

7 while 176 > 0: --- True

8 digit = 176 % 16

digit = 0

9 res = DIGITS[0] + ''

res = '0'

10 number = 176 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + '0'

res = 'B0'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'B0'

4. square\_equal(-4, 26, 90) = [9.0, -2.5]

3 def square\_equal(a=-4, b=26, c=90)

4 if -4 != 0: --- True

5 D = 26\*26 - 4\*-4\*90

D = 2116

6 if 2116 > 0: --- True

7 x1 = (-26 - sqrt(2116)) / (2\*-4)

x1 = 9.0

8 x2 = (-26 + sqrt(2116)) / (2\*-4)

x2 = -2.5

9 return [9.0, -2.5]

5. square\_equal(-44, -8, -76) = []

3 def square\_equal(a=-44, b=-8, c=-76)

4 if -44 != 0: --- True

5 D = -8\*-8 - 4\*-44\*-76

D = -13312

6 if -13312 > 0: --- False

10 elif -13312 == 0: --- False

12 else:

13 return []

6. findmax([-27, -2, 75, 46, -75]) = 75

1 def findmax(items=[-27, -2, 75, 46, -75])

2 if len([-27, -2, 75, 46, -75]) == 0: --- False

4 m = items[0]

m = -27

5 i = 1

6 while 1 < len([-27, -2, 75, 46, -75]): --- True

7 if -27 < items[1]: --- True

8 m = items[1]

m = -2

9 i = 1 + 1

i = 2

6 while 2 < len([-27, -2, 75, 46, -75]): --- True

7 if -2 < items[2]: --- True

8 m = items[2]

m = 75

9 i = 2 + 1

i = 3

6 while 3 < len([-27, -2, 75, 46, -75]): --- True

7 if 75 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-27, -2, 75, 46, -75]): --- True

7 if 75 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-27, -2, 75, 46, -75]): --- False

10 return 75

7. unique([5, 5, 28]) = [5, 28]

1 def unique(items=[5, 5, 28])

2 res = []

3 i = 0

4 while 0 < len([5, 5, 28]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [5]

7 i = 0 + 1

i = 1

4 while 1 < len([5, 5, 28]): --- True

5 if len([5]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([5, 5, 28]): --- True

5 if len([5]) == 0 or res[-1] != items[2]: --- True

6 res = [5] + [items[2]]

res = [5, 28]

7 i = 2 + 1

i = 3

4 while 3 < len([5, 5, 28]): --- False

8 return [5, 28]

8. join(':', [88, 1, 67, 41]) = '88:1:67:41'

1 def join(sep=:, items=[88, 1, 67, 41])

2 res = ''

3 if len([88, 1, 67, 41]) > 0: --- True

4 res = str(items[0])

res = '88'

5 items = items[1:]

items = [1, 67, 41]

6 while len([1, 67, 41]) > 0: --- True

7 res = '88' + ':' + str(items[0])

res = '88:1'

8 items = items[1:]

items = [67, 41]

6 while len([67, 41]) > 0: --- True

7 res = '88:1' + ':' + str(items[0])

res = '88:1:67'

8 items = items[1:]

items = [41]

6 while len([41]) > 0: --- True

7 res = '88:1:67' + ':' + str(items[0])

res = '88:1:67:41'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '88:1:67:41'

Вариант: 2-3-7

1. gcd(-60, -42) = 6

1 def gcd(x=-60, y=-42)

2 if -60 < 0: --- True

3 x = --60

x = 60

4 if -42 < 0: --- True

5 y = --42

y = 42

6 if 60 == 0: --- False

8 while 42 != 0: --- True

9 rem = 60 % 42

rem = 18

10 x = 42

11 y = 18

8 while 18 != 0: --- True

9 rem = 42 % 18

rem = 6

10 x = 18

11 y = 6

8 while 6 != 0: --- True

9 rem = 18 % 6

rem = 0

10 x = 6

11 y = 0

8 while 0 != 0: --- False

12 return 6

2. gcd(0, -73) = 73

1 def gcd(x=0, y=-73)

2 if 0 < 0: --- False

4 if -73 < 0: --- True

5 y = --73

y = 73

6 if 0 == 0: --- True

7 return 73

3. hex(217) = 'D9'

3 def hex(number=217)

4 if 217 == 0: --- False

6 res = ''

7 while 217 > 0: --- True

8 digit = 217 % 16

digit = 9

9 res = DIGITS[9] + ''

res = '9'

10 number = 217 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + '9'

res = 'D9'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'D9'

4. square\_equal(47, -94, 47) = [1.0]

3 def square\_equal(a=47, b=-94, c=47)

4 if 47 != 0: --- True

5 D = -94\*-94 - 4\*47\*47

D = 0

6 if 0 > 0: --- False

10 elif 0 == 0: --- True

11 return [1.0]

5. square\_equal(88, 14, 62) = []

3 def square\_equal(a=88, b=14, c=62)

4 if 88 != 0: --- True

5 D = 14\*14 - 4\*88\*62

D = -21628

6 if -21628 > 0: --- False

10 elif -21628 == 0: --- False

12 else:

13 return []

6. findmax([90, 97, -21, -2, -74, 17]) = 97

1 def findmax(items=[90, 97, -21, -2, -74, 17])

2 if len([90, 97, -21, -2, -74, 17]) == 0: --- False

4 m = items[0]

m = 90

5 i = 1

6 while 1 < len([90, 97, -21, -2, -74, 17]): --- True

7 if 90 < items[1]: --- True

8 m = items[1]

m = 97

9 i = 1 + 1

i = 2

6 while 2 < len([90, 97, -21, -2, -74, 17]): --- True

7 if 97 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([90, 97, -21, -2, -74, 17]): --- True

7 if 97 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([90, 97, -21, -2, -74, 17]): --- True

7 if 97 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([90, 97, -21, -2, -74, 17]): --- True

7 if 97 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([90, 97, -21, -2, -74, 17]): --- False

10 return 97

7. unique([-68, -94, -10, -10]) = [-68, -94, -10]

1 def unique(items=[-68, -94, -10, -10])

2 res = []

3 i = 0

4 while 0 < len([-68, -94, -10, -10]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [-68]

7 i = 0 + 1

i = 1

4 while 1 < len([-68, -94, -10, -10]): --- True

5 if len([-68]) == 0 or res[-1] != items[1]: --- True

6 res = [-68] + [items[1]]

res = [-68, -94]

7 i = 1 + 1

i = 2

4 while 2 < len([-68, -94, -10, -10]): --- True

5 if len([-68, -94]) == 0 or res[-1] != items[2]: --- True

6 res = [-68, -94] + [items[2]]

res = [-68, -94, -10]

7 i = 2 + 1

i = 3

4 while 3 < len([-68, -94, -10, -10]): --- True

5 if len([-68, -94, -10]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([-68, -94, -10, -10]): --- False

8 return [-68, -94, -10]

8. join(':', [55, 78, 1]) = '55:78:1'

1 def join(sep=:, items=[55, 78, 1])

2 res = ''

3 if len([55, 78, 1]) > 0: --- True

4 res = str(items[0])

res = '55'

5 items = items[1:]

items = [78, 1]

6 while len([78, 1]) > 0: --- True

7 res = '55' + ':' + str(items[0])

res = '55:78'

8 items = items[1:]

items = [1]

6 while len([1]) > 0: --- True

7 res = '55:78' + ':' + str(items[0])

res = '55:78:1'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '55:78:1'

Вариант: 2-3-8

1. gcd(81, 21) = 3

1 def gcd(x=81, y=21)

2 if 81 < 0: --- False

4 if 21 < 0: --- False

6 if 81 == 0: --- False

8 while 21 != 0: --- True

9 rem = 81 % 21

rem = 18

10 x = 21

11 y = 18

8 while 18 != 0: --- True

9 rem = 21 % 18

rem = 3

10 x = 18

11 y = 3

8 while 3 != 0: --- True

9 rem = 18 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(39, 0) = 39

1 def gcd(x=39, y=0)

2 if 39 < 0: --- False

4 if 0 < 0: --- False

6 if 39 == 0: --- False

8 while 0 != 0: --- False

12 return 39

3. hex(220) = 'DC'

3 def hex(number=220)

4 if 220 == 0: --- False

6 res = ''

7 while 220 > 0: --- True

8 digit = 220 % 16

digit = 12

9 res = DIGITS[12] + ''

res = 'C'

10 number = 220 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + 'C'

res = 'DC'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'DC'

4. square\_equal(0, -24, -66) = [-2.75]

3 def square\_equal(a=0, b=-24, c=-66)

4 if 0 != 0: --- False

14 else:

15 if -24 != 0: --- True

16 return [-2.75]

5. square\_equal(83, -5, 87) = []

3 def square\_equal(a=83, b=-5, c=87)

4 if 83 != 0: --- True

5 D = -5\*-5 - 4\*83\*87

D = -28859

6 if -28859 > 0: --- False

10 elif -28859 == 0: --- False

12 else:

13 return []

6. findmax([5, 92, 39, 54, 61]) = 92

1 def findmax(items=[5, 92, 39, 54, 61])

2 if len([5, 92, 39, 54, 61]) == 0: --- False

4 m = items[0]

m = 5

5 i = 1

6 while 1 < len([5, 92, 39, 54, 61]): --- True

7 if 5 < items[1]: --- True

8 m = items[1]

m = 92

9 i = 1 + 1

i = 2

6 while 2 < len([5, 92, 39, 54, 61]): --- True

7 if 92 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([5, 92, 39, 54, 61]): --- True

7 if 92 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([5, 92, 39, 54, 61]): --- True

7 if 92 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([5, 92, 39, 54, 61]): --- False

10 return 92

7. unique([64, 64, -32]) = [64, -32]

1 def unique(items=[64, 64, -32])

2 res = []

3 i = 0

4 while 0 < len([64, 64, -32]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [64]

7 i = 0 + 1

i = 1

4 while 1 < len([64, 64, -32]): --- True

5 if len([64]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([64, 64, -32]): --- True

5 if len([64]) == 0 or res[-1] != items[2]: --- True

6 res = [64] + [items[2]]

res = [64, -32]

7 i = 2 + 1

i = 3

4 while 3 < len([64, 64, -32]): --- False

8 return [64, -32]

8. join(',', [27, 70, 44, 82]) = '27,70,44,82'

1 def join(sep=,, items=[27, 70, 44, 82])

2 res = ''

3 if len([27, 70, 44, 82]) > 0: --- True

4 res = str(items[0])

res = '27'

5 items = items[1:]

items = [70, 44, 82]

6 while len([70, 44, 82]) > 0: --- True

7 res = '27' + ',' + str(items[0])

res = '27,70'

8 items = items[1:]

items = [44, 82]

6 while len([44, 82]) > 0: --- True

7 res = '27,70' + ',' + str(items[0])

res = '27,70,44'

8 items = items[1:]

items = [82]

6 while len([82]) > 0: --- True

7 res = '27,70,44' + ',' + str(items[0])

res = '27,70,44,82'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '27,70,44,82'

Вариант: 2-3-9

1. gcd(78, 51) = 3

1 def gcd(x=78, y=51)

2 if 78 < 0: --- False

4 if 51 < 0: --- False

6 if 78 == 0: --- False

8 while 51 != 0: --- True

9 rem = 78 % 51

rem = 27

10 x = 51

11 y = 27

8 while 27 != 0: --- True

9 rem = 51 % 27

rem = 24

10 x = 27

11 y = 24

8 while 24 != 0: --- True

9 rem = 27 % 24

rem = 3

10 x = 24

11 y = 3

8 while 3 != 0: --- True

9 rem = 24 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(0, -3) = 3

1 def gcd(x=0, y=-3)

2 if 0 < 0: --- False

4 if -3 < 0: --- True

5 y = --3

y = 3

6 if 0 == 0: --- True

7 return 3

3. hex(249) = 'F9'

3 def hex(number=249)

4 if 249 == 0: --- False

6 res = ''

7 while 249 > 0: --- True

8 digit = 249 % 16

digit = 9

9 res = DIGITS[9] + ''

res = '9'

10 number = 249 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + '9'

res = 'F9'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'F9'

4. square\_equal(8, 12, 0) = [-1.5, 0.0]

3 def square\_equal(a=8, b=12, c=0)

4 if 8 != 0: --- True

5 D = 12\*12 - 4\*8\*0

D = 144

6 if 144 > 0: --- True

7 x1 = (-12 - sqrt(144)) / (2\*8)

x1 = -1.5

8 x2 = (-12 + sqrt(144)) / (2\*8)

x2 = 0.0

9 return [-1.5, 0.0]

5. square\_equal(7, 12, 82) = []

3 def square\_equal(a=7, b=12, c=82)

4 if 7 != 0: --- True

5 D = 12\*12 - 4\*7\*82

D = -2152

6 if -2152 > 0: --- False

10 elif -2152 == 0: --- False

12 else:

13 return []

6. findmax([83, -71, -23, 60, 80, 3]) = 83

1 def findmax(items=[83, -71, -23, 60, 80, 3])

2 if len([83, -71, -23, 60, 80, 3]) == 0: --- False

4 m = items[0]

m = 83

5 i = 1

6 while 1 < len([83, -71, -23, 60, 80, 3]): --- True

7 if 83 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([83, -71, -23, 60, 80, 3]): --- True

7 if 83 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([83, -71, -23, 60, 80, 3]): --- True

7 if 83 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([83, -71, -23, 60, 80, 3]): --- True

7 if 83 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([83, -71, -23, 60, 80, 3]): --- True

7 if 83 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([83, -71, -23, 60, 80, 3]): --- False

10 return 83

7. unique([79, 89, 89]) = [79, 89]

1 def unique(items=[79, 89, 89])

2 res = []

3 i = 0

4 while 0 < len([79, 89, 89]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [79]

7 i = 0 + 1

i = 1

4 while 1 < len([79, 89, 89]): --- True

5 if len([79]) == 0 or res[-1] != items[1]: --- True

6 res = [79] + [items[1]]

res = [79, 89]

7 i = 1 + 1

i = 2

4 while 2 < len([79, 89, 89]): --- True

5 if len([79, 89]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([79, 89, 89]): --- False

8 return [79, 89]

8. join(';', [1, 75, 27, 9]) = '1;75;27;9'

1 def join(sep=;, items=[1, 75, 27, 9])

2 res = ''

3 if len([1, 75, 27, 9]) > 0: --- True

4 res = str(items[0])

res = '1'

5 items = items[1:]

items = [75, 27, 9]

6 while len([75, 27, 9]) > 0: --- True

7 res = '1' + ';' + str(items[0])

res = '1;75'

8 items = items[1:]

items = [27, 9]

6 while len([27, 9]) > 0: --- True

7 res = '1;75' + ';' + str(items[0])

res = '1;75;27'

8 items = items[1:]

items = [9]

6 while len([9]) > 0: --- True

7 res = '1;75;27' + ';' + str(items[0])

res = '1;75;27;9'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '1;75;27;9'

Вариант: 2-3-10

1. gcd(98, -56) = 14

1 def gcd(x=98, y=-56)

2 if 98 < 0: --- False

4 if -56 < 0: --- True

5 y = --56

y = 56

6 if 98 == 0: --- False

8 while 56 != 0: --- True

9 rem = 98 % 56

rem = 42

10 x = 56

11 y = 42

8 while 42 != 0: --- True

9 rem = 56 % 42

rem = 14

10 x = 42

11 y = 14

8 while 14 != 0: --- True

9 rem = 42 % 14

rem = 0

10 x = 14

11 y = 0

8 while 0 != 0: --- False

12 return 14

2. gcd(-45, 0) = 45

1 def gcd(x=-45, y=0)

2 if -45 < 0: --- True

3 x = --45

x = 45

4 if 0 < 0: --- False

6 if 45 == 0: --- False

8 while 0 != 0: --- False

12 return 45

3. hex(240) = 'F0'

3 def hex(number=240)

4 if 240 == 0: --- False

6 res = ''

7 while 240 > 0: --- True

8 digit = 240 % 16

digit = 0

9 res = DIGITS[0] + ''

res = '0'

10 number = 240 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + '0'

res = 'F0'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'F0'

4. square\_equal(6, -90, 84) = [1.0, 14.0]

3 def square\_equal(a=6, b=-90, c=84)

4 if 6 != 0: --- True

5 D = -90\*-90 - 4\*6\*84

D = 6084

6 if 6084 > 0: --- True

7 x1 = (--90 - sqrt(6084)) / (2\*6)

x1 = 1.0

8 x2 = (--90 + sqrt(6084)) / (2\*6)

x2 = 14.0

9 return [1.0, 14.0]

5. square\_equal(-94, -99, -34) = []

3 def square\_equal(a=-94, b=-99, c=-34)

4 if -94 != 0: --- True

5 D = -99\*-99 - 4\*-94\*-34

D = -2983

6 if -2983 > 0: --- False

10 elif -2983 == 0: --- False

12 else:

13 return []

6. findmax([-52, 52, 67, 36]) = 67

1 def findmax(items=[-52, 52, 67, 36])

2 if len([-52, 52, 67, 36]) == 0: --- False

4 m = items[0]

m = -52

5 i = 1

6 while 1 < len([-52, 52, 67, 36]): --- True

7 if -52 < items[1]: --- True

8 m = items[1]

m = 52

9 i = 1 + 1

i = 2

6 while 2 < len([-52, 52, 67, 36]): --- True

7 if 52 < items[2]: --- True

8 m = items[2]

m = 67

9 i = 2 + 1

i = 3

6 while 3 < len([-52, 52, 67, 36]): --- True

7 if 67 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-52, 52, 67, 36]): --- False

10 return 67

7. unique([74, 74, 15, 74]) = [74, 15, 74]

1 def unique(items=[74, 74, 15, 74])

2 res = []

3 i = 0

4 while 0 < len([74, 74, 15, 74]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [74]

7 i = 0 + 1

i = 1

4 while 1 < len([74, 74, 15, 74]): --- True

5 if len([74]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([74, 74, 15, 74]): --- True

5 if len([74]) == 0 or res[-1] != items[2]: --- True

6 res = [74] + [items[2]]

res = [74, 15]

7 i = 2 + 1

i = 3

4 while 3 < len([74, 74, 15, 74]): --- True

5 if len([74, 15]) == 0 or res[-1] != items[3]: --- True

6 res = [74, 15] + [items[3]]

res = [74, 15, 74]

7 i = 3 + 1

i = 4

4 while 4 < len([74, 74, 15, 74]): --- False

8 return [74, 15, 74]

8. join(',', [10, 69, 56, 79]) = '10,69,56,79'

1 def join(sep=,, items=[10, 69, 56, 79])

2 res = ''

3 if len([10, 69, 56, 79]) > 0: --- True

4 res = str(items[0])

res = '10'

5 items = items[1:]

items = [69, 56, 79]

6 while len([69, 56, 79]) > 0: --- True

7 res = '10' + ',' + str(items[0])

res = '10,69'

8 items = items[1:]

items = [56, 79]

6 while len([56, 79]) > 0: --- True

7 res = '10,69' + ',' + str(items[0])

res = '10,69,56'

8 items = items[1:]

items = [79]

6 while len([79]) > 0: --- True

7 res = '10,69,56' + ',' + str(items[0])

res = '10,69,56,79'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '10,69,56,79'

Вариант: 2-3-11

1. gcd(-80, 68) = 4

1 def gcd(x=-80, y=68)

2 if -80 < 0: --- True

3 x = --80

x = 80

4 if 68 < 0: --- False

6 if 80 == 0: --- False

8 while 68 != 0: --- True

9 rem = 80 % 68

rem = 12

10 x = 68

11 y = 12

8 while 12 != 0: --- True

9 rem = 68 % 12

rem = 8

10 x = 12

11 y = 8

8 while 8 != 0: --- True

9 rem = 12 % 8

rem = 4

10 x = 8

11 y = 4

8 while 4 != 0: --- True

9 rem = 8 % 4

rem = 0

10 x = 4

11 y = 0

8 while 0 != 0: --- False

12 return 4

2. gcd(0, 85) = 85

1 def gcd(x=0, y=85)

2 if 0 < 0: --- False

4 if 85 < 0: --- False

6 if 0 == 0: --- True

7 return 85

3. hex(213) = 'D5'

3 def hex(number=213)

4 if 213 == 0: --- False

6 res = ''

7 while 213 > 0: --- True

8 digit = 213 % 16

digit = 5

9 res = DIGITS[5] + ''

res = '5'

10 number = 213 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + '5'

res = 'D5'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'D5'

4. square\_equal(0, 7, -70) = [10.0]

3 def square\_equal(a=0, b=7, c=-70)

4 if 0 != 0: --- False

14 else:

15 if 7 != 0: --- True

16 return [10.0]

5. square\_equal(-63, -74, -54) = []

3 def square\_equal(a=-63, b=-74, c=-54)

4 if -63 != 0: --- True

5 D = -74\*-74 - 4\*-63\*-54

D = -8132

6 if -8132 > 0: --- False

10 elif -8132 == 0: --- False

12 else:

13 return []

6. findmax([5, 9, -39, 76, 19]) = 76

1 def findmax(items=[5, 9, -39, 76, 19])

2 if len([5, 9, -39, 76, 19]) == 0: --- False

4 m = items[0]

m = 5

5 i = 1

6 while 1 < len([5, 9, -39, 76, 19]): --- True

7 if 5 < items[1]: --- True

8 m = items[1]

m = 9

9 i = 1 + 1

i = 2

6 while 2 < len([5, 9, -39, 76, 19]): --- True

7 if 9 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([5, 9, -39, 76, 19]): --- True

7 if 9 < items[3]: --- True

8 m = items[3]

m = 76

9 i = 3 + 1

i = 4

6 while 4 < len([5, 9, -39, 76, 19]): --- True

7 if 76 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([5, 9, -39, 76, 19]): --- False

10 return 76

7. unique([-91, -91, 42]) = [-91, 42]

1 def unique(items=[-91, -91, 42])

2 res = []

3 i = 0

4 while 0 < len([-91, -91, 42]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [-91]

7 i = 0 + 1

i = 1

4 while 1 < len([-91, -91, 42]): --- True

5 if len([-91]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-91, -91, 42]): --- True

5 if len([-91]) == 0 or res[-1] != items[2]: --- True

6 res = [-91] + [items[2]]

res = [-91, 42]

7 i = 2 + 1

i = 3

4 while 3 < len([-91, -91, 42]): --- False

8 return [-91, 42]

8. join(',', [73, 99, 73, 13]) = '73,99,73,13'

1 def join(sep=,, items=[73, 99, 73, 13])

2 res = ''

3 if len([73, 99, 73, 13]) > 0: --- True

4 res = str(items[0])

res = '73'

5 items = items[1:]

items = [99, 73, 13]

6 while len([99, 73, 13]) > 0: --- True

7 res = '73' + ',' + str(items[0])

res = '73,99'

8 items = items[1:]

items = [73, 13]

6 while len([73, 13]) > 0: --- True

7 res = '73,99' + ',' + str(items[0])

res = '73,99,73'

8 items = items[1:]

items = [13]

6 while len([13]) > 0: --- True

7 res = '73,99,73' + ',' + str(items[0])

res = '73,99,73,13'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '73,99,73,13'

Вариант: 2-3-12

1. gcd(54, -90) = 18

1 def gcd(x=54, y=-90)

2 if 54 < 0: --- False

4 if -90 < 0: --- True

5 y = --90

y = 90

6 if 54 == 0: --- False

8 while 90 != 0: --- True

9 rem = 54 % 90

rem = 54

10 x = 90

11 y = 54

8 while 54 != 0: --- True

9 rem = 90 % 54

rem = 36

10 x = 54

11 y = 36

8 while 36 != 0: --- True

9 rem = 54 % 36

rem = 18

10 x = 36

11 y = 18

8 while 18 != 0: --- True

9 rem = 36 % 18

rem = 0

10 x = 18

11 y = 0

8 while 0 != 0: --- False

12 return 18

2. gcd(0, 5) = 5

1 def gcd(x=0, y=5)

2 if 0 < 0: --- False

4 if 5 < 0: --- False

6 if 0 == 0: --- True

7 return 5

3. hex(185) = 'B9'

3 def hex(number=185)

4 if 185 == 0: --- False

6 res = ''

7 while 185 > 0: --- True

8 digit = 185 % 16

digit = 9

9 res = DIGITS[9] + ''

res = '9'

10 number = 185 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + '9'

res = 'B9'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'B9'

4. square\_equal(-10, 51, -44) = [4.0, 1.1]

3 def square\_equal(a=-10, b=51, c=-44)

4 if -10 != 0: --- True

5 D = 51\*51 - 4\*-10\*-44

D = 841

6 if 841 > 0: --- True

7 x1 = (-51 - sqrt(841)) / (2\*-10)

x1 = 4.0

8 x2 = (-51 + sqrt(841)) / (2\*-10)

x2 = 1.1

9 return [4.0, 1.1]

5. square\_equal(75, -94, 56) = []

3 def square\_equal(a=75, b=-94, c=56)

4 if 75 != 0: --- True

5 D = -94\*-94 - 4\*75\*56

D = -7964

6 if -7964 > 0: --- False

10 elif -7964 == 0: --- False

12 else:

13 return []

6. findmax([92, -76, -69, -5, -19]) = 92

1 def findmax(items=[92, -76, -69, -5, -19])

2 if len([92, -76, -69, -5, -19]) == 0: --- False

4 m = items[0]

m = 92

5 i = 1

6 while 1 < len([92, -76, -69, -5, -19]): --- True

7 if 92 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([92, -76, -69, -5, -19]): --- True

7 if 92 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([92, -76, -69, -5, -19]): --- True

7 if 92 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([92, -76, -69, -5, -19]): --- True

7 if 92 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([92, -76, -69, -5, -19]): --- False

10 return 92

7. unique([49, 49, 61, 49]) = [49, 61, 49]

1 def unique(items=[49, 49, 61, 49])

2 res = []

3 i = 0

4 while 0 < len([49, 49, 61, 49]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [49]

7 i = 0 + 1

i = 1

4 while 1 < len([49, 49, 61, 49]): --- True

5 if len([49]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([49, 49, 61, 49]): --- True

5 if len([49]) == 0 or res[-1] != items[2]: --- True

6 res = [49] + [items[2]]

res = [49, 61]

7 i = 2 + 1

i = 3

4 while 3 < len([49, 49, 61, 49]): --- True

5 if len([49, 61]) == 0 or res[-1] != items[3]: --- True

6 res = [49, 61] + [items[3]]

res = [49, 61, 49]

7 i = 3 + 1

i = 4

4 while 4 < len([49, 49, 61, 49]): --- False

8 return [49, 61, 49]

8. join(';', [72, 14, 64]) = '72;14;64'

1 def join(sep=;, items=[72, 14, 64])

2 res = ''

3 if len([72, 14, 64]) > 0: --- True

4 res = str(items[0])

res = '72'

5 items = items[1:]

items = [14, 64]

6 while len([14, 64]) > 0: --- True

7 res = '72' + ';' + str(items[0])

res = '72;14'

8 items = items[1:]

items = [64]

6 while len([64]) > 0: --- True

7 res = '72;14' + ';' + str(items[0])

res = '72;14;64'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '72;14;64'

Вариант: 2-3-13

1. gcd(-28, -49) = 7

1 def gcd(x=-28, y=-49)

2 if -28 < 0: --- True

3 x = --28

x = 28

4 if -49 < 0: --- True

5 y = --49

y = 49

6 if 28 == 0: --- False

8 while 49 != 0: --- True

9 rem = 28 % 49

rem = 28

10 x = 49

11 y = 28

8 while 28 != 0: --- True

9 rem = 49 % 28

rem = 21

10 x = 28

11 y = 21

8 while 21 != 0: --- True

9 rem = 28 % 21

rem = 7

10 x = 21

11 y = 7

8 while 7 != 0: --- True

9 rem = 21 % 7

rem = 0

10 x = 7

11 y = 0

8 while 0 != 0: --- False

12 return 7

2. gcd(0, 8) = 8

1 def gcd(x=0, y=8)

2 if 0 < 0: --- False

4 if 8 < 0: --- False

6 if 0 == 0: --- True

7 return 8

3. hex(182) = 'B6'

3 def hex(number=182)

4 if 182 == 0: --- False

6 res = ''

7 while 182 > 0: --- True

8 digit = 182 % 16

digit = 6

9 res = DIGITS[6] + ''

res = '6'

10 number = 182 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + '6'

res = 'B6'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'B6'

4. square\_equal(5, 39, 72) = [-4.8, -3.0]

3 def square\_equal(a=5, b=39, c=72)

4 if 5 != 0: --- True

5 D = 39\*39 - 4\*5\*72

D = 81

6 if 81 > 0: --- True

7 x1 = (-39 - sqrt(81)) / (2\*5)

x1 = -4.8

8 x2 = (-39 + sqrt(81)) / (2\*5)

x2 = -3.0

9 return [-4.8, -3.0]

5. square\_equal(-65, 37, -29) = []

3 def square\_equal(a=-65, b=37, c=-29)

4 if -65 != 0: --- True

5 D = 37\*37 - 4\*-65\*-29

D = -6171

6 if -6171 > 0: --- False

10 elif -6171 == 0: --- False

12 else:

13 return []

6. findmax([-52, -42, 33, 59]) = 59

1 def findmax(items=[-52, -42, 33, 59])

2 if len([-52, -42, 33, 59]) == 0: --- False

4 m = items[0]

m = -52

5 i = 1

6 while 1 < len([-52, -42, 33, 59]): --- True

7 if -52 < items[1]: --- True

8 m = items[1]

m = -42

9 i = 1 + 1

i = 2

6 while 2 < len([-52, -42, 33, 59]): --- True

7 if -42 < items[2]: --- True

8 m = items[2]

m = 33

9 i = 2 + 1

i = 3

6 while 3 < len([-52, -42, 33, 59]): --- True

7 if 33 < items[3]: --- True

8 m = items[3]

m = 59

9 i = 3 + 1

i = 4

6 while 4 < len([-52, -42, 33, 59]): --- False

10 return 59

7. unique([91, -2, -2, 91]) = [91, -2, 91]

1 def unique(items=[91, -2, -2, 91])

2 res = []

3 i = 0

4 while 0 < len([91, -2, -2, 91]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [91]

7 i = 0 + 1

i = 1

4 while 1 < len([91, -2, -2, 91]): --- True

5 if len([91]) == 0 or res[-1] != items[1]: --- True

6 res = [91] + [items[1]]

res = [91, -2]

7 i = 1 + 1

i = 2

4 while 2 < len([91, -2, -2, 91]): --- True

5 if len([91, -2]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([91, -2, -2, 91]): --- True

5 if len([91, -2]) == 0 or res[-1] != items[3]: --- True

6 res = [91, -2] + [items[3]]

res = [91, -2, 91]

7 i = 3 + 1

i = 4

4 while 4 < len([91, -2, -2, 91]): --- False

8 return [91, -2, 91]

8. join(';', [77, 55, 8]) = '77;55;8'

1 def join(sep=;, items=[77, 55, 8])

2 res = ''

3 if len([77, 55, 8]) > 0: --- True

4 res = str(items[0])

res = '77'

5 items = items[1:]

items = [55, 8]

6 while len([55, 8]) > 0: --- True

7 res = '77' + ';' + str(items[0])

res = '77;55'

8 items = items[1:]

items = [8]

6 while len([8]) > 0: --- True

7 res = '77;55' + ';' + str(items[0])

res = '77;55;8'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '77;55;8'

Вариант: 2-3-14

1. gcd(57, -51) = 3

1 def gcd(x=57, y=-51)

2 if 57 < 0: --- False

4 if -51 < 0: --- True

5 y = --51

y = 51

6 if 57 == 0: --- False

8 while 51 != 0: --- True

9 rem = 57 % 51

rem = 6

10 x = 51

11 y = 6

8 while 6 != 0: --- True

9 rem = 51 % 6

rem = 3

10 x = 6

11 y = 3

8 while 3 != 0: --- True

9 rem = 6 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(-39, 0) = 39

1 def gcd(x=-39, y=0)

2 if -39 < 0: --- True

3 x = --39

x = 39

4 if 0 < 0: --- False

6 if 39 == 0: --- False

8 while 0 != 0: --- False

12 return 39

3. hex(165) = 'A5'

3 def hex(number=165)

4 if 165 == 0: --- False

6 res = ''

7 while 165 > 0: --- True

8 digit = 165 % 16

digit = 5

9 res = DIGITS[5] + ''

res = '5'

10 number = 165 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + '5'

res = 'A5'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'A5'

4. square\_equal(5, -41, 80) = [3.2, 5.0]

3 def square\_equal(a=5, b=-41, c=80)

4 if 5 != 0: --- True

5 D = -41\*-41 - 4\*5\*80

D = 81

6 if 81 > 0: --- True

7 x1 = (--41 - sqrt(81)) / (2\*5)

x1 = 3.2

8 x2 = (--41 + sqrt(81)) / (2\*5)

x2 = 5.0

9 return [3.2, 5.0]

5. square\_equal(-97, -16, -10) = []

3 def square\_equal(a=-97, b=-16, c=-10)

4 if -97 != 0: --- True

5 D = -16\*-16 - 4\*-97\*-10

D = -3624

6 if -3624 > 0: --- False

10 elif -3624 == 0: --- False

12 else:

13 return []

6. findmax([25, 87, 71, -51]) = 87

1 def findmax(items=[25, 87, 71, -51])

2 if len([25, 87, 71, -51]) == 0: --- False

4 m = items[0]

m = 25

5 i = 1

6 while 1 < len([25, 87, 71, -51]): --- True

7 if 25 < items[1]: --- True

8 m = items[1]

m = 87

9 i = 1 + 1

i = 2

6 while 2 < len([25, 87, 71, -51]): --- True

7 if 87 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([25, 87, 71, -51]): --- True

7 if 87 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([25, 87, 71, -51]): --- False

10 return 87

7. unique([-35, -35, 91]) = [-35, 91]

1 def unique(items=[-35, -35, 91])

2 res = []

3 i = 0

4 while 0 < len([-35, -35, 91]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [-35]

7 i = 0 + 1

i = 1

4 while 1 < len([-35, -35, 91]): --- True

5 if len([-35]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-35, -35, 91]): --- True

5 if len([-35]) == 0 or res[-1] != items[2]: --- True

6 res = [-35] + [items[2]]

res = [-35, 91]

7 i = 2 + 1

i = 3

4 while 3 < len([-35, -35, 91]): --- False

8 return [-35, 91]

8. join('+', [6, 90, 72]) = '6+90+72'

1 def join(sep=+, items=[6, 90, 72])

2 res = ''

3 if len([6, 90, 72]) > 0: --- True

4 res = str(items[0])

res = '6'

5 items = items[1:]

items = [90, 72]

6 while len([90, 72]) > 0: --- True

7 res = '6' + '+' + str(items[0])

res = '6+90'

8 items = items[1:]

items = [72]

6 while len([72]) > 0: --- True

7 res = '6+90' + '+' + str(items[0])

res = '6+90+72'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '6+90+72'

Вариант: 2-3-15

1. gcd(96, -51) = 3

1 def gcd(x=96, y=-51)

2 if 96 < 0: --- False

4 if -51 < 0: --- True

5 y = --51

y = 51

6 if 96 == 0: --- False

8 while 51 != 0: --- True

9 rem = 96 % 51

rem = 45

10 x = 51

11 y = 45

8 while 45 != 0: --- True

9 rem = 51 % 45

rem = 6

10 x = 45

11 y = 6

8 while 6 != 0: --- True

9 rem = 45 % 6

rem = 3

10 x = 6

11 y = 3

8 while 3 != 0: --- True

9 rem = 6 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(86, 0) = 86

1 def gcd(x=86, y=0)

2 if 86 < 0: --- False

4 if 0 < 0: --- False

6 if 86 == 0: --- False

8 while 0 != 0: --- False

12 return 86

3. hex(206) = 'CE'

3 def hex(number=206)

4 if 206 == 0: --- False

6 res = ''

7 while 206 > 0: --- True

8 digit = 206 % 16

digit = 14

9 res = DIGITS[14] + ''

res = 'E'

10 number = 206 // 16

number = 12

7 while 12 > 0: --- True

8 digit = 12 % 16

digit = 12

9 res = DIGITS[12] + 'E'

res = 'CE'

10 number = 12 // 16

number = 0

7 while 0 > 0: --- False

11 return 'CE'

4. square\_equal(20, -36, -23) = [-0.5, 2.3]

3 def square\_equal(a=20, b=-36, c=-23)

4 if 20 != 0: --- True

5 D = -36\*-36 - 4\*20\*-23

D = 3136

6 if 3136 > 0: --- True

7 x1 = (--36 - sqrt(3136)) / (2\*20)

x1 = -0.5

8 x2 = (--36 + sqrt(3136)) / (2\*20)

x2 = 2.3

9 return [-0.5, 2.3]

5. square\_equal(69, -45, 96) = []

3 def square\_equal(a=69, b=-45, c=96)

4 if 69 != 0: --- True

5 D = -45\*-45 - 4\*69\*96

D = -24471

6 if -24471 > 0: --- False

10 elif -24471 == 0: --- False

12 else:

13 return []

6. findmax([-70, 57, -2, -72, -92]) = 57

1 def findmax(items=[-70, 57, -2, -72, -92])

2 if len([-70, 57, -2, -72, -92]) == 0: --- False

4 m = items[0]

m = -70

5 i = 1

6 while 1 < len([-70, 57, -2, -72, -92]): --- True

7 if -70 < items[1]: --- True

8 m = items[1]

m = 57

9 i = 1 + 1

i = 2

6 while 2 < len([-70, 57, -2, -72, -92]): --- True

7 if 57 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-70, 57, -2, -72, -92]): --- True

7 if 57 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-70, 57, -2, -72, -92]): --- True

7 if 57 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-70, 57, -2, -72, -92]): --- False

10 return 57

7. unique([4, 4, 1]) = [4, 1]

1 def unique(items=[4, 4, 1])

2 res = []

3 i = 0

4 while 0 < len([4, 4, 1]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [4]

7 i = 0 + 1

i = 1

4 while 1 < len([4, 4, 1]): --- True

5 if len([4]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([4, 4, 1]): --- True

5 if len([4]) == 0 or res[-1] != items[2]: --- True

6 res = [4] + [items[2]]

res = [4, 1]

7 i = 2 + 1

i = 3

4 while 3 < len([4, 4, 1]): --- False

8 return [4, 1]

8. join(',', [78, 99, 36]) = '78,99,36'

1 def join(sep=,, items=[78, 99, 36])

2 res = ''

3 if len([78, 99, 36]) > 0: --- True

4 res = str(items[0])

res = '78'

5 items = items[1:]

items = [99, 36]

6 while len([99, 36]) > 0: --- True

7 res = '78' + ',' + str(items[0])

res = '78,99'

8 items = items[1:]

items = [36]

6 while len([36]) > 0: --- True

7 res = '78,99' + ',' + str(items[0])

res = '78,99,36'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '78,99,36'

Вариант: 2-3-16

1. gcd(-60, -44) = 4

1 def gcd(x=-60, y=-44)

2 if -60 < 0: --- True

3 x = --60

x = 60

4 if -44 < 0: --- True

5 y = --44

y = 44

6 if 60 == 0: --- False

8 while 44 != 0: --- True

9 rem = 60 % 44

rem = 16

10 x = 44

11 y = 16

8 while 16 != 0: --- True

9 rem = 44 % 16

rem = 12

10 x = 16

11 y = 12

8 while 12 != 0: --- True

9 rem = 16 % 12

rem = 4

10 x = 12

11 y = 4

8 while 4 != 0: --- True

9 rem = 12 % 4

rem = 0

10 x = 4

11 y = 0

8 while 0 != 0: --- False

12 return 4

2. gcd(0, -30) = 30

1 def gcd(x=0, y=-30)

2 if 0 < 0: --- False

4 if -30 < 0: --- True

5 y = --30

y = 30

6 if 0 == 0: --- True

7 return 30

3. hex(234) = 'EA'

3 def hex(number=234)

4 if 234 == 0: --- False

6 res = ''

7 while 234 > 0: --- True

8 digit = 234 % 16

digit = 10

9 res = DIGITS[10] + ''

res = 'A'

10 number = 234 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + 'A'

res = 'EA'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'EA'

4. square\_equal(3, 6, 3) = [-1.0]

3 def square\_equal(a=3, b=6, c=3)

4 if 3 != 0: --- True

5 D = 6\*6 - 4\*3\*3

D = 0

6 if 0 > 0: --- False

10 elif 0 == 0: --- True

11 return [-1.0]

5. square\_equal(-85, 18, -98) = []

3 def square\_equal(a=-85, b=18, c=-98)

4 if -85 != 0: --- True

5 D = 18\*18 - 4\*-85\*-98

D = -32996

6 if -32996 > 0: --- False

10 elif -32996 == 0: --- False

12 else:

13 return []

6. findmax([-19, 72, 15, -3]) = 72

1 def findmax(items=[-19, 72, 15, -3])

2 if len([-19, 72, 15, -3]) == 0: --- False

4 m = items[0]

m = -19

5 i = 1

6 while 1 < len([-19, 72, 15, -3]): --- True

7 if -19 < items[1]: --- True

8 m = items[1]

m = 72

9 i = 1 + 1

i = 2

6 while 2 < len([-19, 72, 15, -3]): --- True

7 if 72 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-19, 72, 15, -3]): --- True

7 if 72 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-19, 72, 15, -3]): --- False

10 return 72

7. unique([25, -34, 37, 37]) = [25, -34, 37]

1 def unique(items=[25, -34, 37, 37])

2 res = []

3 i = 0

4 while 0 < len([25, -34, 37, 37]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [25]

7 i = 0 + 1

i = 1

4 while 1 < len([25, -34, 37, 37]): --- True

5 if len([25]) == 0 or res[-1] != items[1]: --- True

6 res = [25] + [items[1]]

res = [25, -34]

7 i = 1 + 1

i = 2

4 while 2 < len([25, -34, 37, 37]): --- True

5 if len([25, -34]) == 0 or res[-1] != items[2]: --- True

6 res = [25, -34] + [items[2]]

res = [25, -34, 37]

7 i = 2 + 1

i = 3

4 while 3 < len([25, -34, 37, 37]): --- True

5 if len([25, -34, 37]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([25, -34, 37, 37]): --- False

8 return [25, -34, 37]

8. join(':', [52, 91, 4, 51]) = '52:91:4:51'

1 def join(sep=:, items=[52, 91, 4, 51])

2 res = ''

3 if len([52, 91, 4, 51]) > 0: --- True

4 res = str(items[0])

res = '52'

5 items = items[1:]

items = [91, 4, 51]

6 while len([91, 4, 51]) > 0: --- True

7 res = '52' + ':' + str(items[0])

res = '52:91'

8 items = items[1:]

items = [4, 51]

6 while len([4, 51]) > 0: --- True

7 res = '52:91' + ':' + str(items[0])

res = '52:91:4'

8 items = items[1:]

items = [51]

6 while len([51]) > 0: --- True

7 res = '52:91:4' + ':' + str(items[0])

res = '52:91:4:51'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '52:91:4:51'

Вариант: 2-3-17

1. gcd(57, -48) = 3

1 def gcd(x=57, y=-48)

2 if 57 < 0: --- False

4 if -48 < 0: --- True

5 y = --48

y = 48

6 if 57 == 0: --- False

8 while 48 != 0: --- True

9 rem = 57 % 48

rem = 9

10 x = 48

11 y = 9

8 while 9 != 0: --- True

9 rem = 48 % 9

rem = 3

10 x = 9

11 y = 3

8 while 3 != 0: --- True

9 rem = 9 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(-61, 0) = 61

1 def gcd(x=-61, y=0)

2 if -61 < 0: --- True

3 x = --61

x = 61

4 if 0 < 0: --- False

6 if 61 == 0: --- False

8 while 0 != 0: --- False

12 return 61

3. hex(211) = 'D3'

3 def hex(number=211)

4 if 211 == 0: --- False

6 res = ''

7 while 211 > 0: --- True

8 digit = 211 % 16

digit = 3

9 res = DIGITS[3] + ''

res = '3'

10 number = 211 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + '3'

res = 'D3'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'D3'

4. square\_equal(-4, -68, 35) = [0.5, -17.5]

3 def square\_equal(a=-4, b=-68, c=35)

4 if -4 != 0: --- True

5 D = -68\*-68 - 4\*-4\*35

D = 5184

6 if 5184 > 0: --- True

7 x1 = (--68 - sqrt(5184)) / (2\*-4)

x1 = 0.5

8 x2 = (--68 + sqrt(5184)) / (2\*-4)

x2 = -17.5

9 return [0.5, -17.5]

5. square\_equal(71, 59, 37) = []

3 def square\_equal(a=71, b=59, c=37)

4 if 71 != 0: --- True

5 D = 59\*59 - 4\*71\*37

D = -7027

6 if -7027 > 0: --- False

10 elif -7027 == 0: --- False

12 else:

13 return []

6. findmax([68, 14, -34, 88, 77]) = 88

1 def findmax(items=[68, 14, -34, 88, 77])

2 if len([68, 14, -34, 88, 77]) == 0: --- False

4 m = items[0]

m = 68

5 i = 1

6 while 1 < len([68, 14, -34, 88, 77]): --- True

7 if 68 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([68, 14, -34, 88, 77]): --- True

7 if 68 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([68, 14, -34, 88, 77]): --- True

7 if 68 < items[3]: --- True

8 m = items[3]

m = 88

9 i = 3 + 1

i = 4

6 while 4 < len([68, 14, -34, 88, 77]): --- True

7 if 88 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([68, 14, -34, 88, 77]): --- False

10 return 88

7. unique([-67, -67, 85]) = [-67, 85]

1 def unique(items=[-67, -67, 85])

2 res = []

3 i = 0

4 while 0 < len([-67, -67, 85]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [-67]

7 i = 0 + 1

i = 1

4 while 1 < len([-67, -67, 85]): --- True

5 if len([-67]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-67, -67, 85]): --- True

5 if len([-67]) == 0 or res[-1] != items[2]: --- True

6 res = [-67] + [items[2]]

res = [-67, 85]

7 i = 2 + 1

i = 3

4 while 3 < len([-67, -67, 85]): --- False

8 return [-67, 85]

8. join(',', [23, 32, 37, 72]) = '23,32,37,72'

1 def join(sep=,, items=[23, 32, 37, 72])

2 res = ''

3 if len([23, 32, 37, 72]) > 0: --- True

4 res = str(items[0])

res = '23'

5 items = items[1:]

items = [32, 37, 72]

6 while len([32, 37, 72]) > 0: --- True

7 res = '23' + ',' + str(items[0])

res = '23,32'

8 items = items[1:]

items = [37, 72]

6 while len([37, 72]) > 0: --- True

7 res = '23,32' + ',' + str(items[0])

res = '23,32,37'

8 items = items[1:]

items = [72]

6 while len([72]) > 0: --- True

7 res = '23,32,37' + ',' + str(items[0])

res = '23,32,37,72'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '23,32,37,72'

Вариант: 2-3-18

1. gcd(-84, -56) = 28

1 def gcd(x=-84, y=-56)

2 if -84 < 0: --- True

3 x = --84

x = 84

4 if -56 < 0: --- True

5 y = --56

y = 56

6 if 84 == 0: --- False

8 while 56 != 0: --- True

9 rem = 84 % 56

rem = 28

10 x = 56

11 y = 28

8 while 28 != 0: --- True

9 rem = 56 % 28

rem = 0

10 x = 28

11 y = 0

8 while 0 != 0: --- False

12 return 28

2. gcd(-46, 0) = 46

1 def gcd(x=-46, y=0)

2 if -46 < 0: --- True

3 x = --46

x = 46

4 if 0 < 0: --- False

6 if 46 == 0: --- False

8 while 0 != 0: --- False

12 return 46

3. hex(195) = 'C3'

3 def hex(number=195)

4 if 195 == 0: --- False

6 res = ''

7 while 195 > 0: --- True

8 digit = 195 % 16

digit = 3

9 res = DIGITS[3] + ''

res = '3'

10 number = 195 // 16

number = 12

7 while 12 > 0: --- True

8 digit = 12 % 16

digit = 12

9 res = DIGITS[12] + '3'

res = 'C3'

10 number = 12 // 16

number = 0

7 while 0 > 0: --- False

11 return 'C3'

4. square\_equal(0, 100, 26) = [-0.26]

3 def square\_equal(a=0, b=100, c=26)

4 if 0 != 0: --- False

14 else:

15 if 100 != 0: --- True

16 return [-0.26]

5. square\_equal(-34, -95, -75) = []

3 def square\_equal(a=-34, b=-95, c=-75)

4 if -34 != 0: --- True

5 D = -95\*-95 - 4\*-34\*-75

D = -1175

6 if -1175 > 0: --- False

10 elif -1175 == 0: --- False

12 else:

13 return []

6. findmax([33, -46, 80, 17, -70, -72]) = 80

1 def findmax(items=[33, -46, 80, 17, -70, -72])

2 if len([33, -46, 80, 17, -70, -72]) == 0: --- False

4 m = items[0]

m = 33

5 i = 1

6 while 1 < len([33, -46, 80, 17, -70, -72]): --- True

7 if 33 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([33, -46, 80, 17, -70, -72]): --- True

7 if 33 < items[2]: --- True

8 m = items[2]

m = 80

9 i = 2 + 1

i = 3

6 while 3 < len([33, -46, 80, 17, -70, -72]): --- True

7 if 80 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([33, -46, 80, 17, -70, -72]): --- True

7 if 80 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([33, -46, 80, 17, -70, -72]): --- True

7 if 80 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([33, -46, 80, 17, -70, -72]): --- False

10 return 80

7. unique([-63, 91, -63, -63]) = [-63, 91, -63]

1 def unique(items=[-63, 91, -63, -63])

2 res = []

3 i = 0

4 while 0 < len([-63, 91, -63, -63]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [-63]

7 i = 0 + 1

i = 1

4 while 1 < len([-63, 91, -63, -63]): --- True

5 if len([-63]) == 0 or res[-1] != items[1]: --- True

6 res = [-63] + [items[1]]

res = [-63, 91]

7 i = 1 + 1

i = 2

4 while 2 < len([-63, 91, -63, -63]): --- True

5 if len([-63, 91]) == 0 or res[-1] != items[2]: --- True

6 res = [-63, 91] + [items[2]]

res = [-63, 91, -63]

7 i = 2 + 1

i = 3

4 while 3 < len([-63, 91, -63, -63]): --- True

5 if len([-63, 91, -63]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([-63, 91, -63, -63]): --- False

8 return [-63, 91, -63]

8. join(';', [36, 91, 50, 16]) = '36;91;50;16'

1 def join(sep=;, items=[36, 91, 50, 16])

2 res = ''

3 if len([36, 91, 50, 16]) > 0: --- True

4 res = str(items[0])

res = '36'

5 items = items[1:]

items = [91, 50, 16]

6 while len([91, 50, 16]) > 0: --- True

7 res = '36' + ';' + str(items[0])

res = '36;91'

8 items = items[1:]

items = [50, 16]

6 while len([50, 16]) > 0: --- True

7 res = '36;91' + ';' + str(items[0])

res = '36;91;50'

8 items = items[1:]

items = [16]

6 while len([16]) > 0: --- True

7 res = '36;91;50' + ';' + str(items[0])

res = '36;91;50;16'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '36;91;50;16'

Вариант: 2-3-19

1. gcd(-6, -96) = 6

1 def gcd(x=-6, y=-96)

2 if -6 < 0: --- True

3 x = --6

x = 6

4 if -96 < 0: --- True

5 y = --96

y = 96

6 if 6 == 0: --- False

8 while 96 != 0: --- True

9 rem = 6 % 96

rem = 6

10 x = 96

11 y = 6

8 while 6 != 0: --- True

9 rem = 96 % 6

rem = 0

10 x = 6

11 y = 0

8 while 0 != 0: --- False

12 return 6

2. gcd(-34, 0) = 34

1 def gcd(x=-34, y=0)

2 if -34 < 0: --- True

3 x = --34

x = 34

4 if 0 < 0: --- False

6 if 34 == 0: --- False

8 while 0 != 0: --- False

12 return 34

3. hex(230) = 'E6'

3 def hex(number=230)

4 if 230 == 0: --- False

6 res = ''

7 while 230 > 0: --- True

8 digit = 230 % 16

digit = 6

9 res = DIGITS[6] + ''

res = '6'

10 number = 230 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + '6'

res = 'E6'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'E6'

4. square\_equal(5, 98, 0) = [-19.6, 0.0]

3 def square\_equal(a=5, b=98, c=0)

4 if 5 != 0: --- True

5 D = 98\*98 - 4\*5\*0

D = 9604

6 if 9604 > 0: --- True

7 x1 = (-98 - sqrt(9604)) / (2\*5)

x1 = -19.6

8 x2 = (-98 + sqrt(9604)) / (2\*5)

x2 = 0.0

9 return [-19.6, 0.0]

5. square\_equal(-58, -59, -83) = []

3 def square\_equal(a=-58, b=-59, c=-83)

4 if -58 != 0: --- True

5 D = -59\*-59 - 4\*-58\*-83

D = -15775

6 if -15775 > 0: --- False

10 elif -15775 == 0: --- False

12 else:

13 return []

6. findmax([-15, -22, 13, 36]) = 36

1 def findmax(items=[-15, -22, 13, 36])

2 if len([-15, -22, 13, 36]) == 0: --- False

4 m = items[0]

m = -15

5 i = 1

6 while 1 < len([-15, -22, 13, 36]): --- True

7 if -15 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([-15, -22, 13, 36]): --- True

7 if -15 < items[2]: --- True

8 m = items[2]

m = 13

9 i = 2 + 1

i = 3

6 while 3 < len([-15, -22, 13, 36]): --- True

7 if 13 < items[3]: --- True

8 m = items[3]

m = 36

9 i = 3 + 1

i = 4

6 while 4 < len([-15, -22, 13, 36]): --- False

10 return 36

7. unique([7, 7, 35]) = [7, 35]

1 def unique(items=[7, 7, 35])

2 res = []

3 i = 0

4 while 0 < len([7, 7, 35]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [7]

7 i = 0 + 1

i = 1

4 while 1 < len([7, 7, 35]): --- True

5 if len([7]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([7, 7, 35]): --- True

5 if len([7]) == 0 or res[-1] != items[2]: --- True

6 res = [7] + [items[2]]

res = [7, 35]

7 i = 2 + 1

i = 3

4 while 3 < len([7, 7, 35]): --- False

8 return [7, 35]

8. join(':', [66, 67, 99]) = '66:67:99'

1 def join(sep=:, items=[66, 67, 99])

2 res = ''

3 if len([66, 67, 99]) > 0: --- True

4 res = str(items[0])

res = '66'

5 items = items[1:]

items = [67, 99]

6 while len([67, 99]) > 0: --- True

7 res = '66' + ':' + str(items[0])

res = '66:67'

8 items = items[1:]

items = [99]

6 while len([99]) > 0: --- True

7 res = '66:67' + ':' + str(items[0])

res = '66:67:99'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '66:67:99'

Вариант: 2-3-20

1. gcd(-12, 66) = 6

1 def gcd(x=-12, y=66)

2 if -12 < 0: --- True

3 x = --12

x = 12

4 if 66 < 0: --- False

6 if 12 == 0: --- False

8 while 66 != 0: --- True

9 rem = 12 % 66

rem = 12

10 x = 66

11 y = 12

8 while 12 != 0: --- True

9 rem = 66 % 12

rem = 6

10 x = 12

11 y = 6

8 while 6 != 0: --- True

9 rem = 12 % 6

rem = 0

10 x = 6

11 y = 0

8 while 0 != 0: --- False

12 return 6

2. gcd(0, 52) = 52

1 def gcd(x=0, y=52)

2 if 0 < 0: --- False

4 if 52 < 0: --- False

6 if 0 == 0: --- True

7 return 52

3. hex(189) = 'BD'

3 def hex(number=189)

4 if 189 == 0: --- False

6 res = ''

7 while 189 > 0: --- True

8 digit = 189 % 16

digit = 13

9 res = DIGITS[13] + ''

res = 'D'

10 number = 189 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + 'D'

res = 'BD'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'BD'

4. square\_equal(4, 28, -72) = [-9.0, 2.0]

3 def square\_equal(a=4, b=28, c=-72)

4 if 4 != 0: --- True

5 D = 28\*28 - 4\*4\*-72

D = 1936

6 if 1936 > 0: --- True

7 x1 = (-28 - sqrt(1936)) / (2\*4)

x1 = -9.0

8 x2 = (-28 + sqrt(1936)) / (2\*4)

x2 = 2.0

9 return [-9.0, 2.0]

5. square\_equal(49, -26, 43) = []

3 def square\_equal(a=49, b=-26, c=43)

4 if 49 != 0: --- True

5 D = -26\*-26 - 4\*49\*43

D = -7752

6 if -7752 > 0: --- False

10 elif -7752 == 0: --- False

12 else:

13 return []

6. findmax([62, 89, -50, -92, -73]) = 89

1 def findmax(items=[62, 89, -50, -92, -73])

2 if len([62, 89, -50, -92, -73]) == 0: --- False

4 m = items[0]

m = 62

5 i = 1

6 while 1 < len([62, 89, -50, -92, -73]): --- True

7 if 62 < items[1]: --- True

8 m = items[1]

m = 89

9 i = 1 + 1

i = 2

6 while 2 < len([62, 89, -50, -92, -73]): --- True

7 if 89 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([62, 89, -50, -92, -73]): --- True

7 if 89 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([62, 89, -50, -92, -73]): --- True

7 if 89 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([62, 89, -50, -92, -73]): --- False

10 return 89

7. unique([46, 46, 13]) = [46, 13]

1 def unique(items=[46, 46, 13])

2 res = []

3 i = 0

4 while 0 < len([46, 46, 13]): --- True

5 if len([]) == 0 or res[-1] != items[0]: --- True

6 res = [] + [items[0]]

res = [46]

7 i = 0 + 1

i = 1

4 while 1 < len([46, 46, 13]): --- True

5 if len([46]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([46, 46, 13]): --- True

5 if len([46]) == 0 or res[-1] != items[2]: --- True

6 res = [46] + [items[2]]

res = [46, 13]

7 i = 2 + 1

i = 3

4 while 3 < len([46, 46, 13]): --- False

8 return [46, 13]

8. join('+', [68, 63, 47, 16]) = '68+63+47+16'

1 def join(sep=+, items=[68, 63, 47, 16])

2 res = ''

3 if len([68, 63, 47, 16]) > 0: --- True

4 res = str(items[0])

res = '68'

5 items = items[1:]

items = [63, 47, 16]

6 while len([63, 47, 16]) > 0: --- True

7 res = '68' + '+' + str(items[0])

res = '68+63'

8 items = items[1:]

items = [47, 16]

6 while len([47, 16]) > 0: --- True

7 res = '68+63' + '+' + str(items[0])

res = '68+63+47'

8 items = items[1:]

items = [16]

6 while len([16]) > 0: --- True

7 res = '68+63+47' + '+' + str(items[0])

res = '68+63+47+16'

8 items = items[1:]

items = []

6 while len([]) > 0: --- False

9 return '68+63+47+16'