Вариант: 2-1-1

1. gcd(78, -51) = 3

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

2 if 78 < 0: --- False

4 if -51 < 0: --- True

5 y = --51

y = 51

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(-62, 0) = 62

1 def gcd(x=-62, y=0)

2 if -62 < 0: --- True

3 x = --62

x = 62

4 if 0 < 0: --- False

6 if 62 == 0: --- False

8 while 0 != 0: --- False

12 return 62

3. hex(254) = 'FE'

3 def hex(number=254)

4 if 254 == 0: --- False

6 res = ''

7 while 254 > 0: --- True

8 digit = 254 % 16

digit = 14

9 res = DIGITS[14] + ''

res = 'E'

10 number = 254 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + 'E'

res = 'FE'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'FE'

4. square\_equal(-1, -13, -22) = [-2.0, -11.0]

3 def square\_equal(a=-1, b=-13, c=-22)

4 if -1 != 0: --- True

5 D = -13\*-13 - 4\*-1\*-22

D = 81

6 if 81 > 0: --- True

7 x1 = (--13 - sqrt(81)) / (2\*-1)

x1 = -2.0

8 x2 = (--13 + sqrt(81)) / (2\*-1)

x2 = -11.0

9 return [-2.0, -11.0]

5. square\_equal(-37, 12, -48) = []

3 def square\_equal(a=-37, b=12, c=-48)

4 if -37 != 0: --- True

5 D = 12\*12 - 4\*-37\*-48

D = -6960

6 if -6960 > 0: --- False

10 elif -6960 == 0: --- False

12 else:

13 return []

6. findmax([39, -84, -60, 58]) = 58

1 def findmax(items=[39, -84, -60, 58])

2 if len([39, -84, -60, 58]) == 0: --- False

4 m = items[0]

m = 39

5 i = 1

6 while 1 < len([39, -84, -60, 58]): --- True

7 if 39 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([39, -84, -60, 58]): --- True

7 if 39 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([39, -84, -60, 58]): --- True

7 if 39 < items[3]: --- True

8 m = items[3]

m = 58

9 i = 3 + 1

i = 4

6 while 4 < len([39, -84, -60, 58]): --- False

10 return 58

7. unique([-36, -36, -57]) = [-36, -57]

1 def unique(items=[-36, -36, -57])

2 res = []

3 i = 0

4 while 0 < len([-36, -36, -57]): --- True

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

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

res = [-36]

7 i = 0 + 1

i = 1

4 while 1 < len([-36, -36, -57]): --- True

5 if len([-36]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-36, -36, -57]): --- True

5 if len([-36]) == 0 or res[-1] != items[2]: --- True

6 res = [-36] + [items[2]]

res = [-36, -57]

7 i = 2 + 1

i = 3

4 while 3 < len([-36, -36, -57]): --- False

8 return [-36, -57]

8. join('+', [83, 43, 43]) = '83+43+43'

1 def join(sep=+, items=[83, 43, 43])

2 res = ''

3 if len([83, 43, 43]) > 0: --- True

4 res = str(items[0])

res = '83'

5 items = items[1:]

items = [43, 43]

6 while len([43, 43]) > 0: --- True

7 res = '83' + '+' + str(items[0])

res = '83+43'

8 items = items[1:]

items = [43]

6 while len([43]) > 0: --- True

7 res = '83+43' + '+' + str(items[0])

res = '83+43+43'

8 items = items[1:]

items = []

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

9 return '83+43+43'

Вариант: 2-1-2

1. gcd(69, -92) = 23

1 def gcd(x=69, y=-92)

2 if 69 < 0: --- False

4 if -92 < 0: --- True

5 y = --92

y = 92

6 if 69 == 0: --- False

8 while 92 != 0: --- True

9 rem = 69 % 92

rem = 69

10 x = 92

11 y = 69

8 while 69 != 0: --- True

9 rem = 92 % 69

rem = 23

10 x = 69

11 y = 23

8 while 23 != 0: --- True

9 rem = 69 % 23

rem = 0

10 x = 23

11 y = 0

8 while 0 != 0: --- False

12 return 23

2. gcd(-15, 0) = 15

1 def gcd(x=-15, y=0)

2 if -15 < 0: --- True

3 x = --15

x = 15

4 if 0 < 0: --- False

6 if 15 == 0: --- False

8 while 0 != 0: --- False

12 return 15

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(0, -75, 45) = [0.6]

3 def square\_equal(a=0, b=-75, c=45)

4 if 0 != 0: --- False

14 else:

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

16 return [0.6]

5. square\_equal(-85, -55, -50) = []

3 def square\_equal(a=-85, b=-55, c=-50)

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

5 D = -55\*-55 - 4\*-85\*-50

D = -13975

6 if -13975 > 0: --- False

10 elif -13975 == 0: --- False

12 else:

13 return []

6. findmax([-84, -82, -60, -7]) = -7

1 def findmax(items=[-84, -82, -60, -7])

2 if len([-84, -82, -60, -7]) == 0: --- False

4 m = items[0]

m = -84

5 i = 1

6 while 1 < len([-84, -82, -60, -7]): --- True

7 if -84 < items[1]: --- True

8 m = items[1]

m = -82

9 i = 1 + 1

i = 2

6 while 2 < len([-84, -82, -60, -7]): --- True

7 if -82 < items[2]: --- True

8 m = items[2]

m = -60

9 i = 2 + 1

i = 3

6 while 3 < len([-84, -82, -60, -7]): --- True

7 if -60 < items[3]: --- True

8 m = items[3]

m = -7

9 i = 3 + 1

i = 4

6 while 4 < len([-84, -82, -60, -7]): --- False

10 return -7

7. unique([44, 44, 16, 93]) = [44, 16, 93]

1 def unique(items=[44, 44, 16, 93])

2 res = []

3 i = 0

4 while 0 < len([44, 44, 16, 93]): --- True

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

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

res = [44]

7 i = 0 + 1

i = 1

4 while 1 < len([44, 44, 16, 93]): --- True

5 if len([44]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([44, 44, 16, 93]): --- True

5 if len([44]) == 0 or res[-1] != items[2]: --- True

6 res = [44] + [items[2]]

res = [44, 16]

7 i = 2 + 1

i = 3

4 while 3 < len([44, 44, 16, 93]): --- True

5 if len([44, 16]) == 0 or res[-1] != items[3]: --- True

6 res = [44, 16] + [items[3]]

res = [44, 16, 93]

7 i = 3 + 1

i = 4

4 while 4 < len([44, 44, 16, 93]): --- False

8 return [44, 16, 93]

8. join(',', [44, 68, 36]) = '44,68,36'

1 def join(sep=,, items=[44, 68, 36])

2 res = ''

3 if len([44, 68, 36]) > 0: --- True

4 res = str(items[0])

res = '44'

5 items = items[1:]

items = [68, 36]

6 while len([68, 36]) > 0: --- True

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

res = '44,68'

8 items = items[1:]

items = [36]

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

7 res = '44,68' + ',' + str(items[0])

res = '44,68,36'

8 items = items[1:]

items = []

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

9 return '44,68,36'

Вариант: 2-1-3

1. gcd(-63, 66) = 3

1 def gcd(x=-63, y=66)

2 if -63 < 0: --- True

3 x = --63

x = 63

4 if 66 < 0: --- False

6 if 63 == 0: --- False

8 while 66 != 0: --- True

9 rem = 63 % 66

rem = 63

10 x = 66

11 y = 63

8 while 63 != 0: --- True

9 rem = 66 % 63

rem = 3

10 x = 63

11 y = 3

8 while 3 != 0: --- True

9 rem = 63 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(12, 0) = 12

1 def gcd(x=12, y=0)

2 if 12 < 0: --- False

4 if 0 < 0: --- False

6 if 12 == 0: --- False

8 while 0 != 0: --- False

12 return 12

3. hex(244) = 'F4'

3 def hex(number=244)

4 if 244 == 0: --- False

6 res = ''

7 while 244 > 0: --- True

8 digit = 244 % 16

digit = 4

9 res = DIGITS[4] + ''

res = '4'

10 number = 244 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + '4'

res = 'F4'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'F4'

4. square\_equal(-20, 28, 24) = [2.0, -0.6]

3 def square\_equal(a=-20, b=28, c=24)

4 if -20 != 0: --- True

5 D = 28\*28 - 4\*-20\*24

D = 2704

6 if 2704 > 0: --- True

7 x1 = (-28 - sqrt(2704)) / (2\*-20)

x1 = 2.0

8 x2 = (-28 + sqrt(2704)) / (2\*-20)

x2 = -0.6

9 return [2.0, -0.6]

5. square\_equal(-44, 89, -61) = []

3 def square\_equal(a=-44, b=89, c=-61)

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

5 D = 89\*89 - 4\*-44\*-61

D = -2815

6 if -2815 > 0: --- False

10 elif -2815 == 0: --- False

12 else:

13 return []

6. findmax([16, 75, 33, -43, -54]) = 75

1 def findmax(items=[16, 75, 33, -43, -54])

2 if len([16, 75, 33, -43, -54]) == 0: --- False

4 m = items[0]

m = 16

5 i = 1

6 while 1 < len([16, 75, 33, -43, -54]): --- True

7 if 16 < items[1]: --- True

8 m = items[1]

m = 75

9 i = 1 + 1

i = 2

6 while 2 < len([16, 75, 33, -43, -54]): --- True

7 if 75 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([16, 75, 33, -43, -54]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([16, 75, 33, -43, -54]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([16, 75, 33, -43, -54]): --- False

10 return 75

7. unique([-72, 49, -72, -72]) = [-72, 49, -72]

1 def unique(items=[-72, 49, -72, -72])

2 res = []

3 i = 0

4 while 0 < len([-72, 49, -72, -72]): --- True

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

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

res = [-72]

7 i = 0 + 1

i = 1

4 while 1 < len([-72, 49, -72, -72]): --- True

5 if len([-72]) == 0 or res[-1] != items[1]: --- True

6 res = [-72] + [items[1]]

res = [-72, 49]

7 i = 1 + 1

i = 2

4 while 2 < len([-72, 49, -72, -72]): --- True

5 if len([-72, 49]) == 0 or res[-1] != items[2]: --- True

6 res = [-72, 49] + [items[2]]

res = [-72, 49, -72]

7 i = 2 + 1

i = 3

4 while 3 < len([-72, 49, -72, -72]): --- True

5 if len([-72, 49, -72]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([-72, 49, -72, -72]): --- False

8 return [-72, 49, -72]

8. join('+', [61, 21, 28, 54]) = '61+21+28+54'

1 def join(sep=+, items=[61, 21, 28, 54])

2 res = ''

3 if len([61, 21, 28, 54]) > 0: --- True

4 res = str(items[0])

res = '61'

5 items = items[1:]

items = [21, 28, 54]

6 while len([21, 28, 54]) > 0: --- True

7 res = '61' + '+' + str(items[0])

res = '61+21'

8 items = items[1:]

items = [28, 54]

6 while len([28, 54]) > 0: --- True

7 res = '61+21' + '+' + str(items[0])

res = '61+21+28'

8 items = items[1:]

items = [54]

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

7 res = '61+21+28' + '+' + str(items[0])

res = '61+21+28+54'

8 items = items[1:]

items = []

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

9 return '61+21+28+54'

Вариант: 2-1-4

1. gcd(42, 56) = 14

1 def gcd(x=42, y=56)

2 if 42 < 0: --- False

4 if 56 < 0: --- False

6 if 42 == 0: --- False

8 while 56 != 0: --- True

9 rem = 42 % 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(-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(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(4, -69, 0) = [0.0, 17.25]

3 def square\_equal(a=4, b=-69, c=0)

4 if 4 != 0: --- True

5 D = -69\*-69 - 4\*4\*0

D = 4761

6 if 4761 > 0: --- True

7 x1 = (--69 - sqrt(4761)) / (2\*4)

x1 = 0.0

8 x2 = (--69 + sqrt(4761)) / (2\*4)

x2 = 17.25

9 return [0.0, 17.25]

5. square\_equal(-98, 18, -8) = []

3 def square\_equal(a=-98, b=18, c=-8)

4 if -98 != 0: --- True

5 D = 18\*18 - 4\*-98\*-8

D = -2812

6 if -2812 > 0: --- False

10 elif -2812 == 0: --- False

12 else:

13 return []

6. findmax([42, 82, 45, -45, -52]) = 82

1 def findmax(items=[42, 82, 45, -45, -52])

2 if len([42, 82, 45, -45, -52]) == 0: --- False

4 m = items[0]

m = 42

5 i = 1

6 while 1 < len([42, 82, 45, -45, -52]): --- True

7 if 42 < items[1]: --- True

8 m = items[1]

m = 82

9 i = 1 + 1

i = 2

6 while 2 < len([42, 82, 45, -45, -52]): --- True

7 if 82 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([42, 82, 45, -45, -52]): --- True

7 if 82 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([42, 82, 45, -45, -52]): --- True

7 if 82 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([42, 82, 45, -45, -52]): --- False

10 return 82

7. unique([59, 59, -51]) = [59, -51]

1 def unique(items=[59, 59, -51])

2 res = []

3 i = 0

4 while 0 < len([59, 59, -51]): --- True

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

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

res = [59]

7 i = 0 + 1

i = 1

4 while 1 < len([59, 59, -51]): --- True

5 if len([59]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([59, 59, -51]): --- True

5 if len([59]) == 0 or res[-1] != items[2]: --- True

6 res = [59] + [items[2]]

res = [59, -51]

7 i = 2 + 1

i = 3

4 while 3 < len([59, 59, -51]): --- False

8 return [59, -51]

8. join(';', [95, 7, 38, 15]) = '95;7;38;15'

1 def join(sep=;, items=[95, 7, 38, 15])

2 res = ''

3 if len([95, 7, 38, 15]) > 0: --- True

4 res = str(items[0])

res = '95'

5 items = items[1:]

items = [7, 38, 15]

6 while len([7, 38, 15]) > 0: --- True

7 res = '95' + ';' + str(items[0])

res = '95;7'

8 items = items[1:]

items = [38, 15]

6 while len([38, 15]) > 0: --- True

7 res = '95;7' + ';' + str(items[0])

res = '95;7;38'

8 items = items[1:]

items = [15]

6 while len([15]) > 0: --- True

7 res = '95;7;38' + ';' + str(items[0])

res = '95;7;38;15'

8 items = items[1:]

items = []

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

9 return '95;7;38;15'

Вариант: 2-1-5

1. gcd(-20, -48) = 4

1 def gcd(x=-20, y=-48)

2 if -20 < 0: --- True

3 x = --20

x = 20

4 if -48 < 0: --- True

5 y = --48

y = 48

6 if 20 == 0: --- False

8 while 48 != 0: --- True

9 rem = 20 % 48

rem = 20

10 x = 48

11 y = 20

8 while 20 != 0: --- True

9 rem = 48 % 20

rem = 8

10 x = 20

11 y = 8

8 while 8 != 0: --- True

9 rem = 20 % 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(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(247) = 'F7'

3 def hex(number=247)

4 if 247 == 0: --- False

6 res = ''

7 while 247 > 0: --- True

8 digit = 247 % 16

digit = 7

9 res = DIGITS[7] + ''

res = '7'

10 number = 247 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + '7'

res = 'F7'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'F7'

4. square\_equal(2, 29, 39) = [-13.0, -1.5]

3 def square\_equal(a=2, b=29, c=39)

4 if 2 != 0: --- True

5 D = 29\*29 - 4\*2\*39

D = 529

6 if 529 > 0: --- True

7 x1 = (-29 - sqrt(529)) / (2\*2)

x1 = -13.0

8 x2 = (-29 + sqrt(529)) / (2\*2)

x2 = -1.5

9 return [-13.0, -1.5]

5. square\_equal(-60, -66, -59) = []

3 def square\_equal(a=-60, b=-66, c=-59)

4 if -60 != 0: --- True

5 D = -66\*-66 - 4\*-60\*-59

D = -9804

6 if -9804 > 0: --- False

10 elif -9804 == 0: --- False

12 else:

13 return []

6. findmax([3, 36, 24, 25, 25, -45]) = 36

1 def findmax(items=[3, 36, 24, 25, 25, -45])

2 if len([3, 36, 24, 25, 25, -45]) == 0: --- False

4 m = items[0]

m = 3

5 i = 1

6 while 1 < len([3, 36, 24, 25, 25, -45]): --- True

7 if 3 < items[1]: --- True

8 m = items[1]

m = 36

9 i = 1 + 1

i = 2

6 while 2 < len([3, 36, 24, 25, 25, -45]): --- True

7 if 36 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([3, 36, 24, 25, 25, -45]): --- True

7 if 36 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([3, 36, 24, 25, 25, -45]): --- True

7 if 36 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([3, 36, 24, 25, 25, -45]): --- True

7 if 36 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([3, 36, 24, 25, 25, -45]): --- False

10 return 36

7. unique([50, 50, -30]) = [50, -30]

1 def unique(items=[50, 50, -30])

2 res = []

3 i = 0

4 while 0 < len([50, 50, -30]): --- True

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

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

res = [50]

7 i = 0 + 1

i = 1

4 while 1 < len([50, 50, -30]): --- True

5 if len([50]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([50, 50, -30]): --- True

5 if len([50]) == 0 or res[-1] != items[2]: --- True

6 res = [50] + [items[2]]

res = [50, -30]

7 i = 2 + 1

i = 3

4 while 3 < len([50, 50, -30]): --- False

8 return [50, -30]

8. join('+', [74, 17, 94]) = '74+17+94'

1 def join(sep=+, items=[74, 17, 94])

2 res = ''

3 if len([74, 17, 94]) > 0: --- True

4 res = str(items[0])

res = '74'

5 items = items[1:]

items = [17, 94]

6 while len([17, 94]) > 0: --- True

7 res = '74' + '+' + str(items[0])

res = '74+17'

8 items = items[1:]

items = [94]

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

7 res = '74+17' + '+' + str(items[0])

res = '74+17+94'

8 items = items[1:]

items = []

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

9 return '74+17+94'

Вариант: 2-1-6

1. gcd(-68, -8) = 4

1 def gcd(x=-68, y=-8)

2 if -68 < 0: --- True

3 x = --68

x = 68

4 if -8 < 0: --- True

5 y = --8

y = 8

6 if 68 == 0: --- False

8 while 8 != 0: --- True

9 rem = 68 % 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, -44) = 44

1 def gcd(x=0, y=-44)

2 if 0 < 0: --- False

4 if -44 < 0: --- True

5 y = --44

y = 44

6 if 0 == 0: --- True

7 return 44

3. hex(236) = 'EC'

3 def hex(number=236)

4 if 236 == 0: --- False

6 res = ''

7 while 236 > 0: --- True

8 digit = 236 % 16

digit = 12

9 res = DIGITS[12] + ''

res = 'C'

10 number = 236 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + 'C'

res = 'EC'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'EC'

4. square\_equal(40, 46, 0) = [-1.15, 0.0]

3 def square\_equal(a=40, b=46, c=0)

4 if 40 != 0: --- True

5 D = 46\*46 - 4\*40\*0

D = 2116

6 if 2116 > 0: --- True

7 x1 = (-46 - sqrt(2116)) / (2\*40)

x1 = -1.15

8 x2 = (-46 + sqrt(2116)) / (2\*40)

x2 = 0.0

9 return [-1.15, 0.0]

5. square\_equal(49, 35, 40) = []

3 def square\_equal(a=49, b=35, c=40)

4 if 49 != 0: --- True

5 D = 35\*35 - 4\*49\*40

D = -6615

6 if -6615 > 0: --- False

10 elif -6615 == 0: --- False

12 else:

13 return []

6. findmax([-13, -58, 78, -8, -37]) = 78

1 def findmax(items=[-13, -58, 78, -8, -37])

2 if len([-13, -58, 78, -8, -37]) == 0: --- False

4 m = items[0]

m = -13

5 i = 1

6 while 1 < len([-13, -58, 78, -8, -37]): --- True

7 if -13 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([-13, -58, 78, -8, -37]): --- True

7 if -13 < items[2]: --- True

8 m = items[2]

m = 78

9 i = 2 + 1

i = 3

6 while 3 < len([-13, -58, 78, -8, -37]): --- True

7 if 78 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-13, -58, 78, -8, -37]): --- True

7 if 78 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-13, -58, 78, -8, -37]): --- False

10 return 78

7. unique([-50, -50, -17]) = [-50, -17]

1 def unique(items=[-50, -50, -17])

2 res = []

3 i = 0

4 while 0 < len([-50, -50, -17]): --- True

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

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

res = [-50]

7 i = 0 + 1

i = 1

4 while 1 < len([-50, -50, -17]): --- True

5 if len([-50]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-50, -50, -17]): --- True

5 if len([-50]) == 0 or res[-1] != items[2]: --- True

6 res = [-50] + [items[2]]

res = [-50, -17]

7 i = 2 + 1

i = 3

4 while 3 < len([-50, -50, -17]): --- False

8 return [-50, -17]

8. join(';', [90, 4, 33, 73]) = '90;4;33;73'

1 def join(sep=;, items=[90, 4, 33, 73])

2 res = ''

3 if len([90, 4, 33, 73]) > 0: --- True

4 res = str(items[0])

res = '90'

5 items = items[1:]

items = [4, 33, 73]

6 while len([4, 33, 73]) > 0: --- True

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

res = '90;4'

8 items = items[1:]

items = [33, 73]

6 while len([33, 73]) > 0: --- True

7 res = '90;4' + ';' + str(items[0])

res = '90;4;33'

8 items = items[1:]

items = [73]

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

7 res = '90;4;33' + ';' + str(items[0])

res = '90;4;33;73'

8 items = items[1:]

items = []

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

9 return '90;4;33;73'

Вариант: 2-1-7

1. gcd(-80, 100) = 20

1 def gcd(x=-80, y=100)

2 if -80 < 0: --- True

3 x = --80

x = 80

4 if 100 < 0: --- False

6 if 80 == 0: --- False

8 while 100 != 0: --- True

9 rem = 80 % 100

rem = 80

10 x = 100

11 y = 80

8 while 80 != 0: --- True

9 rem = 100 % 80

rem = 20

10 x = 80

11 y = 20

8 while 20 != 0: --- True

9 rem = 80 % 20

rem = 0

10 x = 20

11 y = 0

8 while 0 != 0: --- False

12 return 20

2. gcd(-96, 0) = 96

1 def gcd(x=-96, y=0)

2 if -96 < 0: --- True

3 x = --96

x = 96

4 if 0 < 0: --- False

6 if 96 == 0: --- False

8 while 0 != 0: --- False

12 return 96

3. hex(208) = 'D0'

3 def hex(number=208)

4 if 208 == 0: --- False

6 res = ''

7 while 208 > 0: --- True

8 digit = 208 % 16

digit = 0

9 res = DIGITS[0] + ''

res = '0'

10 number = 208 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + '0'

res = 'D0'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'D0'

4. square\_equal(-4, 16, -16) = [2.0]

3 def square\_equal(a=-4, b=16, c=-16)

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

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

D = 0

6 if 0 > 0: --- False

10 elif 0 == 0: --- True

11 return [2.0]

5. square\_equal(-39, -24, -56) = []

3 def square\_equal(a=-39, b=-24, c=-56)

4 if -39 != 0: --- True

5 D = -24\*-24 - 4\*-39\*-56

D = -8160

6 if -8160 > 0: --- False

10 elif -8160 == 0: --- False

12 else:

13 return []

6. findmax([72, 22, 75, 87]) = 87

1 def findmax(items=[72, 22, 75, 87])

2 if len([72, 22, 75, 87]) == 0: --- False

4 m = items[0]

m = 72

5 i = 1

6 while 1 < len([72, 22, 75, 87]): --- True

7 if 72 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([72, 22, 75, 87]): --- True

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

8 m = items[2]

m = 75

9 i = 2 + 1

i = 3

6 while 3 < len([72, 22, 75, 87]): --- True

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

8 m = items[3]

m = 87

9 i = 3 + 1

i = 4

6 while 4 < len([72, 22, 75, 87]): --- False

10 return 87

7. unique([-63, -63, 80]) = [-63, 80]

1 def unique(items=[-63, -63, 80])

2 res = []

3 i = 0

4 while 0 < len([-63, -63, 80]): --- 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, -63, 80]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([-63, -63, 80]): --- True

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

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

res = [-63, 80]

7 i = 2 + 1

i = 3

4 while 3 < len([-63, -63, 80]): --- False

8 return [-63, 80]

8. join('+', [23, 43, 96, 53]) = '23+43+96+53'

1 def join(sep=+, items=[23, 43, 96, 53])

2 res = ''

3 if len([23, 43, 96, 53]) > 0: --- True

4 res = str(items[0])

res = '23'

5 items = items[1:]

items = [43, 96, 53]

6 while len([43, 96, 53]) > 0: --- True

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

res = '23+43'

8 items = items[1:]

items = [96, 53]

6 while len([96, 53]) > 0: --- True

7 res = '23+43' + '+' + str(items[0])

res = '23+43+96'

8 items = items[1:]

items = [53]

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

7 res = '23+43+96' + '+' + str(items[0])

res = '23+43+96+53'

8 items = items[1:]

items = []

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

9 return '23+43+96+53'

Вариант: 2-1-8

1. gcd(27, -90) = 9

1 def gcd(x=27, y=-90)

2 if 27 < 0: --- False

4 if -90 < 0: --- True

5 y = --90

y = 90

6 if 27 == 0: --- False

8 while 90 != 0: --- True

9 rem = 27 % 90

rem = 27

10 x = 90

11 y = 27

8 while 27 != 0: --- True

9 rem = 90 % 27

rem = 9

10 x = 27

11 y = 9

8 while 9 != 0: --- True

9 rem = 27 % 9

rem = 0

10 x = 9

11 y = 0

8 while 0 != 0: --- False

12 return 9

2. gcd(-20, 0) = 20

1 def gcd(x=-20, y=0)

2 if -20 < 0: --- True

3 x = --20

x = 20

4 if 0 < 0: --- False

6 if 20 == 0: --- False

8 while 0 != 0: --- False

12 return 20

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(40, -68, 28) = [0.7, 1.0]

3 def square\_equal(a=40, b=-68, c=28)

4 if 40 != 0: --- True

5 D = -68\*-68 - 4\*40\*28

D = 144

6 if 144 > 0: --- True

7 x1 = (--68 - sqrt(144)) / (2\*40)

x1 = 0.7

8 x2 = (--68 + sqrt(144)) / (2\*40)

x2 = 1.0

9 return [0.7, 1.0]

5. square\_equal(15, -52, 74) = []

3 def square\_equal(a=15, b=-52, c=74)

4 if 15 != 0: --- True

5 D = -52\*-52 - 4\*15\*74

D = -1736

6 if -1736 > 0: --- False

10 elif -1736 == 0: --- False

12 else:

13 return []

6. findmax([56, 51, 68, -32]) = 68

1 def findmax(items=[56, 51, 68, -32])

2 if len([56, 51, 68, -32]) == 0: --- False

4 m = items[0]

m = 56

5 i = 1

6 while 1 < len([56, 51, 68, -32]): --- True

7 if 56 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([56, 51, 68, -32]): --- True

7 if 56 < items[2]: --- True

8 m = items[2]

m = 68

9 i = 2 + 1

i = 3

6 while 3 < len([56, 51, 68, -32]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([56, 51, 68, -32]): --- False

10 return 68

7. unique([-69, 67, 67]) = [-69, 67]

1 def unique(items=[-69, 67, 67])

2 res = []

3 i = 0

4 while 0 < len([-69, 67, 67]): --- True

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

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

res = [-69]

7 i = 0 + 1

i = 1

4 while 1 < len([-69, 67, 67]): --- True

5 if len([-69]) == 0 or res[-1] != items[1]: --- True

6 res = [-69] + [items[1]]

res = [-69, 67]

7 i = 1 + 1

i = 2

4 while 2 < len([-69, 67, 67]): --- True

5 if len([-69, 67]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-69, 67, 67]): --- False

8 return [-69, 67]

8. join(',', [89, 27, 94]) = '89,27,94'

1 def join(sep=,, items=[89, 27, 94])

2 res = ''

3 if len([89, 27, 94]) > 0: --- True

4 res = str(items[0])

res = '89'

5 items = items[1:]

items = [27, 94]

6 while len([27, 94]) > 0: --- True

7 res = '89' + ',' + str(items[0])

res = '89,27'

8 items = items[1:]

items = [94]

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

7 res = '89,27' + ',' + str(items[0])

res = '89,27,94'

8 items = items[1:]

items = []

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

9 return '89,27,94'

Вариант: 2-1-9

1. gcd(-21, -56) = 7

1 def gcd(x=-21, y=-56)

2 if -21 < 0: --- True

3 x = --21

x = 21

4 if -56 < 0: --- True

5 y = --56

y = 56

6 if 21 == 0: --- False

8 while 56 != 0: --- True

9 rem = 21 % 56

rem = 21

10 x = 56

11 y = 21

8 while 21 != 0: --- True

9 rem = 56 % 21

rem = 14

10 x = 21

11 y = 14

8 while 14 != 0: --- True

9 rem = 21 % 14

rem = 7

10 x = 14

11 y = 7

8 while 7 != 0: --- True

9 rem = 14 % 7

rem = 0

10 x = 7

11 y = 0

8 while 0 != 0: --- False

12 return 7

2. gcd(-72, 0) = 72

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

2 if -72 < 0: --- True

3 x = --72

x = 72

4 if 0 < 0: --- False

6 if 72 == 0: --- False

8 while 0 != 0: --- False

12 return 72

3. hex(226) = 'E2'

3 def hex(number=226)

4 if 226 == 0: --- False

6 res = ''

7 while 226 > 0: --- True

8 digit = 226 % 16

digit = 2

9 res = DIGITS[2] + ''

res = '2'

10 number = 226 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + '2'

res = 'E2'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'E2'

4. square\_equal(0, -25, -76) = [-3.04]

3 def square\_equal(a=0, b=-25, c=-76)

4 if 0 != 0: --- False

14 else:

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

16 return [-3.04]

5. square\_equal(46, -43, 58) = []

3 def square\_equal(a=46, b=-43, c=58)

4 if 46 != 0: --- True

5 D = -43\*-43 - 4\*46\*58

D = -8823

6 if -8823 > 0: --- False

10 elif -8823 == 0: --- False

12 else:

13 return []

6. findmax([59, -89, 86, 90]) = 90

1 def findmax(items=[59, -89, 86, 90])

2 if len([59, -89, 86, 90]) == 0: --- False

4 m = items[0]

m = 59

5 i = 1

6 while 1 < len([59, -89, 86, 90]): --- True

7 if 59 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([59, -89, 86, 90]): --- True

7 if 59 < items[2]: --- True

8 m = items[2]

m = 86

9 i = 2 + 1

i = 3

6 while 3 < len([59, -89, 86, 90]): --- True

7 if 86 < items[3]: --- True

8 m = items[3]

m = 90

9 i = 3 + 1

i = 4

6 while 4 < len([59, -89, 86, 90]): --- False

10 return 90

7. unique([-10, -10, 33]) = [-10, 33]

1 def unique(items=[-10, -10, 33])

2 res = []

3 i = 0

4 while 0 < len([-10, -10, 33]): --- True

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

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

res = [-10]

7 i = 0 + 1

i = 1

4 while 1 < len([-10, -10, 33]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([-10, -10, 33]): --- True

5 if len([-10]) == 0 or res[-1] != items[2]: --- True

6 res = [-10] + [items[2]]

res = [-10, 33]

7 i = 2 + 1

i = 3

4 while 3 < len([-10, -10, 33]): --- False

8 return [-10, 33]

8. join(':', [93, 75, 73]) = '93:75:73'

1 def join(sep=:, items=[93, 75, 73])

2 res = ''

3 if len([93, 75, 73]) > 0: --- True

4 res = str(items[0])

res = '93'

5 items = items[1:]

items = [75, 73]

6 while len([75, 73]) > 0: --- True

7 res = '93' + ':' + str(items[0])

res = '93:75'

8 items = items[1:]

items = [73]

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

7 res = '93:75' + ':' + str(items[0])

res = '93:75:73'

8 items = items[1:]

items = []

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

9 return '93:75:73'

Вариант: 2-1-10

1. gcd(15, -48) = 3

1 def gcd(x=15, y=-48)

2 if 15 < 0: --- False

4 if -48 < 0: --- True

5 y = --48

y = 48

6 if 15 == 0: --- False

8 while 48 != 0: --- True

9 rem = 15 % 48

rem = 15

10 x = 48

11 y = 15

8 while 15 != 0: --- True

9 rem = 48 % 15

rem = 3

10 x = 15

11 y = 3

8 while 3 != 0: --- True

9 rem = 15 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(-79, 0) = 79

1 def gcd(x=-79, y=0)

2 if -79 < 0: --- True

3 x = --79

x = 79

4 if 0 < 0: --- False

6 if 79 == 0: --- False

8 while 0 != 0: --- False

12 return 79

3. hex(188) = 'BC'

3 def hex(number=188)

4 if 188 == 0: --- False

6 res = ''

7 while 188 > 0: --- True

8 digit = 188 % 16

digit = 12

9 res = DIGITS[12] + ''

res = 'C'

10 number = 188 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + 'C'

res = 'BC'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'BC'

4. square\_equal(1, -73, 72) = [1.0, 72.0]

3 def square\_equal(a=1, b=-73, c=72)

4 if 1 != 0: --- True

5 D = -73\*-73 - 4\*1\*72

D = 5041

6 if 5041 > 0: --- True

7 x1 = (--73 - sqrt(5041)) / (2\*1)

x1 = 1.0

8 x2 = (--73 + sqrt(5041)) / (2\*1)

x2 = 72.0

9 return [1.0, 72.0]

5. square\_equal(-13, -3, -32) = []

3 def square\_equal(a=-13, b=-3, c=-32)

4 if -13 != 0: --- True

5 D = -3\*-3 - 4\*-13\*-32

D = -1655

6 if -1655 > 0: --- False

10 elif -1655 == 0: --- False

12 else:

13 return []

6. findmax([35, 93, 30, 60, -70]) = 93

1 def findmax(items=[35, 93, 30, 60, -70])

2 if len([35, 93, 30, 60, -70]) == 0: --- False

4 m = items[0]

m = 35

5 i = 1

6 while 1 < len([35, 93, 30, 60, -70]): --- True

7 if 35 < items[1]: --- True

8 m = items[1]

m = 93

9 i = 1 + 1

i = 2

6 while 2 < len([35, 93, 30, 60, -70]): --- True

7 if 93 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([35, 93, 30, 60, -70]): --- True

7 if 93 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([35, 93, 30, 60, -70]): --- True

7 if 93 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([35, 93, 30, 60, -70]): --- False

10 return 93

7. unique([-89, -89, -63, 36]) = [-89, -63, 36]

1 def unique(items=[-89, -89, -63, 36])

2 res = []

3 i = 0

4 while 0 < len([-89, -89, -63, 36]): --- True

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

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

res = [-89]

7 i = 0 + 1

i = 1

4 while 1 < len([-89, -89, -63, 36]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([-89, -89, -63, 36]): --- True

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

6 res = [-89] + [items[2]]

res = [-89, -63]

7 i = 2 + 1

i = 3

4 while 3 < len([-89, -89, -63, 36]): --- True

5 if len([-89, -63]) == 0 or res[-1] != items[3]: --- True

6 res = [-89, -63] + [items[3]]

res = [-89, -63, 36]

7 i = 3 + 1

i = 4

4 while 4 < len([-89, -89, -63, 36]): --- False

8 return [-89, -63, 36]

8. join('+', [44, 52, 14]) = '44+52+14'

1 def join(sep=+, items=[44, 52, 14])

2 res = ''

3 if len([44, 52, 14]) > 0: --- True

4 res = str(items[0])

res = '44'

5 items = items[1:]

items = [52, 14]

6 while len([52, 14]) > 0: --- True

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

res = '44+52'

8 items = items[1:]

items = [14]

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

7 res = '44+52' + '+' + str(items[0])

res = '44+52+14'

8 items = items[1:]

items = []

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

9 return '44+52+14'

Вариант: 2-1-11

1. gcd(63, -36) = 9

1 def gcd(x=63, y=-36)

2 if 63 < 0: --- False

4 if -36 < 0: --- True

5 y = --36

y = 36

6 if 63 == 0: --- False

8 while 36 != 0: --- True

9 rem = 63 % 36

rem = 27

10 x = 36

11 y = 27

8 while 27 != 0: --- True

9 rem = 36 % 27

rem = 9

10 x = 27

11 y = 9

8 while 9 != 0: --- True

9 rem = 27 % 9

rem = 0

10 x = 9

11 y = 0

8 while 0 != 0: --- False

12 return 9

2. gcd(0, 15) = 15

1 def gcd(x=0, y=15)

2 if 0 < 0: --- False

4 if 15 < 0: --- False

6 if 0 == 0: --- True

7 return 15

3. hex(184) = 'B8'

3 def hex(number=184)

4 if 184 == 0: --- False

6 res = ''

7 while 184 > 0: --- True

8 digit = 184 % 16

digit = 8

9 res = DIGITS[8] + ''

res = '8'

10 number = 184 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + '8'

res = 'B8'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'B8'

4. square\_equal(24, -90, 39) = [0.5, 3.25]

3 def square\_equal(a=24, b=-90, c=39)

4 if 24 != 0: --- True

5 D = -90\*-90 - 4\*24\*39

D = 4356

6 if 4356 > 0: --- True

7 x1 = (--90 - sqrt(4356)) / (2\*24)

x1 = 0.5

8 x2 = (--90 + sqrt(4356)) / (2\*24)

x2 = 3.25

9 return [0.5, 3.25]

5. square\_equal(9, 6, 8) = []

3 def square\_equal(a=9, b=6, c=8)

4 if 9 != 0: --- True

5 D = 6\*6 - 4\*9\*8

D = -252

6 if -252 > 0: --- False

10 elif -252 == 0: --- False

12 else:

13 return []

6. findmax([12, -42, 40, 44]) = 44

1 def findmax(items=[12, -42, 40, 44])

2 if len([12, -42, 40, 44]) == 0: --- False

4 m = items[0]

m = 12

5 i = 1

6 while 1 < len([12, -42, 40, 44]): --- True

7 if 12 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([12, -42, 40, 44]): --- True

7 if 12 < items[2]: --- True

8 m = items[2]

m = 40

9 i = 2 + 1

i = 3

6 while 3 < len([12, -42, 40, 44]): --- True

7 if 40 < items[3]: --- True

8 m = items[3]

m = 44

9 i = 3 + 1

i = 4

6 while 4 < len([12, -42, 40, 44]): --- False

10 return 44

7. unique([-44, -44, -39, 78]) = [-44, -39, 78]

1 def unique(items=[-44, -44, -39, 78])

2 res = []

3 i = 0

4 while 0 < len([-44, -44, -39, 78]): --- True

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

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

res = [-44]

7 i = 0 + 1

i = 1

4 while 1 < len([-44, -44, -39, 78]): --- True

5 if len([-44]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-44, -44, -39, 78]): --- True

5 if len([-44]) == 0 or res[-1] != items[2]: --- True

6 res = [-44] + [items[2]]

res = [-44, -39]

7 i = 2 + 1

i = 3

4 while 3 < len([-44, -44, -39, 78]): --- True

5 if len([-44, -39]) == 0 or res[-1] != items[3]: --- True

6 res = [-44, -39] + [items[3]]

res = [-44, -39, 78]

7 i = 3 + 1

i = 4

4 while 4 < len([-44, -44, -39, 78]): --- False

8 return [-44, -39, 78]

8. join(',', [41, 60, 92]) = '41,60,92'

1 def join(sep=,, items=[41, 60, 92])

2 res = ''

3 if len([41, 60, 92]) > 0: --- True

4 res = str(items[0])

res = '41'

5 items = items[1:]

items = [60, 92]

6 while len([60, 92]) > 0: --- True

7 res = '41' + ',' + str(items[0])

res = '41,60'

8 items = items[1:]

items = [92]

6 while len([92]) > 0: --- True

7 res = '41,60' + ',' + str(items[0])

res = '41,60,92'

8 items = items[1:]

items = []

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

9 return '41,60,92'

Вариант: 2-1-12

1. gcd(-64, -60) = 4

1 def gcd(x=-64, y=-60)

2 if -64 < 0: --- True

3 x = --64

x = 64

4 if -60 < 0: --- True

5 y = --60

y = 60

6 if 64 == 0: --- False

8 while 60 != 0: --- True

9 rem = 64 % 60

rem = 4

10 x = 60

11 y = 4

8 while 4 != 0: --- True

9 rem = 60 % 4

rem = 0

10 x = 4

11 y = 0

8 while 0 != 0: --- False

12 return 4

2. gcd(77, 0) = 77

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

2 if 77 < 0: --- False

4 if 0 < 0: --- False

6 if 77 == 0: --- False

8 while 0 != 0: --- False

12 return 77

3. hex(252) = 'FC'

3 def hex(number=252)

4 if 252 == 0: --- False

6 res = ''

7 while 252 > 0: --- True

8 digit = 252 % 16

digit = 12

9 res = DIGITS[12] + ''

res = 'C'

10 number = 252 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + 'C'

res = 'FC'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'FC'

4. square\_equal(1, 47, -98) = [-49.0, 2.0]

3 def square\_equal(a=1, b=47, c=-98)

4 if 1 != 0: --- True

5 D = 47\*47 - 4\*1\*-98

D = 2601

6 if 2601 > 0: --- True

7 x1 = (-47 - sqrt(2601)) / (2\*1)

x1 = -49.0

8 x2 = (-47 + sqrt(2601)) / (2\*1)

x2 = 2.0

9 return [-49.0, 2.0]

5. square\_equal(80, 8, 40) = []

3 def square\_equal(a=80, b=8, c=40)

4 if 80 != 0: --- True

5 D = 8\*8 - 4\*80\*40

D = -12736

6 if -12736 > 0: --- False

10 elif -12736 == 0: --- False

12 else:

13 return []

6. findmax([-30, 95, -20, -95, 25, -91]) = 95

1 def findmax(items=[-30, 95, -20, -95, 25, -91])

2 if len([-30, 95, -20, -95, 25, -91]) == 0: --- False

4 m = items[0]

m = -30

5 i = 1

6 while 1 < len([-30, 95, -20, -95, 25, -91]): --- True

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

8 m = items[1]

m = 95

9 i = 1 + 1

i = 2

6 while 2 < len([-30, 95, -20, -95, 25, -91]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([-30, 95, -20, -95, 25, -91]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([-30, 95, -20, -95, 25, -91]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([-30, 95, -20, -95, 25, -91]): --- True

7 if 95 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([-30, 95, -20, -95, 25, -91]): --- False

10 return 95

7. unique([-77, -77, -6]) = [-77, -6]

1 def unique(items=[-77, -77, -6])

2 res = []

3 i = 0

4 while 0 < len([-77, -77, -6]): --- True

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

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

res = [-77]

7 i = 0 + 1

i = 1

4 while 1 < len([-77, -77, -6]): --- True

5 if len([-77]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-77, -77, -6]): --- True

5 if len([-77]) == 0 or res[-1] != items[2]: --- True

6 res = [-77] + [items[2]]

res = [-77, -6]

7 i = 2 + 1

i = 3

4 while 3 < len([-77, -77, -6]): --- False

8 return [-77, -6]

8. join(';', [47, 38, 25]) = '47;38;25'

1 def join(sep=;, items=[47, 38, 25])

2 res = ''

3 if len([47, 38, 25]) > 0: --- True

4 res = str(items[0])

res = '47'

5 items = items[1:]

items = [38, 25]

6 while len([38, 25]) > 0: --- True

7 res = '47' + ';' + str(items[0])

res = '47;38'

8 items = items[1:]

items = [25]

6 while len([25]) > 0: --- True

7 res = '47;38' + ';' + str(items[0])

res = '47;38;25'

8 items = items[1:]

items = []

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

9 return '47;38;25'

Вариант: 2-1-13

1. gcd(85, 90) = 5

1 def gcd(x=85, y=90)

2 if 85 < 0: --- False

4 if 90 < 0: --- False

6 if 85 == 0: --- False

8 while 90 != 0: --- True

9 rem = 85 % 90

rem = 85

10 x = 90

11 y = 85

8 while 85 != 0: --- True

9 rem = 90 % 85

rem = 5

10 x = 85

11 y = 5

8 while 5 != 0: --- True

9 rem = 85 % 5

rem = 0

10 x = 5

11 y = 0

8 while 0 != 0: --- False

12 return 5

2. gcd(-52, 0) = 52

1 def gcd(x=-52, y=0)

2 if -52 < 0: --- True

3 x = --52

x = 52

4 if 0 < 0: --- False

6 if 52 == 0: --- False

8 while 0 != 0: --- False

12 return 52

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(-2, -26, -24) = [-1.0, -12.0]

3 def square\_equal(a=-2, b=-26, c=-24)

4 if -2 != 0: --- True

5 D = -26\*-26 - 4\*-2\*-24

D = 484

6 if 484 > 0: --- True

7 x1 = (--26 - sqrt(484)) / (2\*-2)

x1 = -1.0

8 x2 = (--26 + sqrt(484)) / (2\*-2)

x2 = -12.0

9 return [-1.0, -12.0]

5. square\_equal(-94, 87, -72) = []

3 def square\_equal(a=-94, b=87, c=-72)

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

5 D = 87\*87 - 4\*-94\*-72

D = -19503

6 if -19503 > 0: --- False

10 elif -19503 == 0: --- False

12 else:

13 return []

6. findmax([42, 24, -98, 97, -35, 74]) = 97

1 def findmax(items=[42, 24, -98, 97, -35, 74])

2 if len([42, 24, -98, 97, -35, 74]) == 0: --- False

4 m = items[0]

m = 42

5 i = 1

6 while 1 < len([42, 24, -98, 97, -35, 74]): --- True

7 if 42 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([42, 24, -98, 97, -35, 74]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([42, 24, -98, 97, -35, 74]): --- True

7 if 42 < items[3]: --- True

8 m = items[3]

m = 97

9 i = 3 + 1

i = 4

6 while 4 < len([42, 24, -98, 97, -35, 74]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([42, 24, -98, 97, -35, 74]): --- True

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

9 i = 5 + 1

i = 6

6 while 6 < len([42, 24, -98, 97, -35, 74]): --- False

10 return 97

7. unique([-57, -57, 1, -57]) = [-57, 1, -57]

1 def unique(items=[-57, -57, 1, -57])

2 res = []

3 i = 0

4 while 0 < len([-57, -57, 1, -57]): --- True

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

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

res = [-57]

7 i = 0 + 1

i = 1

4 while 1 < len([-57, -57, 1, -57]): --- True

5 if len([-57]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-57, -57, 1, -57]): --- True

5 if len([-57]) == 0 or res[-1] != items[2]: --- True

6 res = [-57] + [items[2]]

res = [-57, 1]

7 i = 2 + 1

i = 3

4 while 3 < len([-57, -57, 1, -57]): --- True

5 if len([-57, 1]) == 0 or res[-1] != items[3]: --- True

6 res = [-57, 1] + [items[3]]

res = [-57, 1, -57]

7 i = 3 + 1

i = 4

4 while 4 < len([-57, -57, 1, -57]): --- False

8 return [-57, 1, -57]

8. join(':', [77, 39, 57, 95]) = '77:39:57:95'

1 def join(sep=:, items=[77, 39, 57, 95])

2 res = ''

3 if len([77, 39, 57, 95]) > 0: --- True

4 res = str(items[0])

res = '77'

5 items = items[1:]

items = [39, 57, 95]

6 while len([39, 57, 95]) > 0: --- True

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

res = '77:39'

8 items = items[1:]

items = [57, 95]

6 while len([57, 95]) > 0: --- True

7 res = '77:39' + ':' + str(items[0])

res = '77:39:57'

8 items = items[1:]

items = [95]

6 while len([95]) > 0: --- True

7 res = '77:39:57' + ':' + str(items[0])

res = '77:39:57:95'

8 items = items[1:]

items = []

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

9 return '77:39:57:95'

Вариант: 2-1-14

1. gcd(-81, 90) = 9

1 def gcd(x=-81, y=90)

2 if -81 < 0: --- True

3 x = --81

x = 81

4 if 90 < 0: --- False

6 if 81 == 0: --- False

8 while 90 != 0: --- True

9 rem = 81 % 90

rem = 81

10 x = 90

11 y = 81

8 while 81 != 0: --- True

9 rem = 90 % 81

rem = 9

10 x = 81

11 y = 9

8 while 9 != 0: --- True

9 rem = 81 % 9

rem = 0

10 x = 9

11 y = 0

8 while 0 != 0: --- False

12 return 9

2. gcd(0, -82) = 82

1 def gcd(x=0, y=-82)

2 if 0 < 0: --- False

4 if -82 < 0: --- True

5 y = --82

y = 82

6 if 0 == 0: --- True

7 return 82

3. hex(178) = 'B2'

3 def hex(number=178)

4 if 178 == 0: --- False

6 res = ''

7 while 178 > 0: --- True

8 digit = 178 % 16

digit = 2

9 res = DIGITS[2] + ''

res = '2'

10 number = 178 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + '2'

res = 'B2'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'B2'

4. square\_equal(-4, 62, 66) = [16.5, -1.0]

3 def square\_equal(a=-4, b=62, c=66)

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

5 D = 62\*62 - 4\*-4\*66

D = 4900

6 if 4900 > 0: --- True

7 x1 = (-62 - sqrt(4900)) / (2\*-4)

x1 = 16.5

8 x2 = (-62 + sqrt(4900)) / (2\*-4)

x2 = -1.0

9 return [16.5, -1.0]

5. square\_equal(-52, -5, -61) = []

3 def square\_equal(a=-52, b=-5, c=-61)

4 if -52 != 0: --- True

5 D = -5\*-5 - 4\*-52\*-61

D = -12663

6 if -12663 > 0: --- False

10 elif -12663 == 0: --- False

12 else:

13 return []

6. findmax([-53, 54, 35, 93, -50]) = 93

1 def findmax(items=[-53, 54, 35, 93, -50])

2 if len([-53, 54, 35, 93, -50]) == 0: --- False

4 m = items[0]

m = -53

5 i = 1

6 while 1 < len([-53, 54, 35, 93, -50]): --- True

7 if -53 < items[1]: --- True

8 m = items[1]

m = 54

9 i = 1 + 1

i = 2

6 while 2 < len([-53, 54, 35, 93, -50]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([-53, 54, 35, 93, -50]): --- True

7 if 54 < items[3]: --- True

8 m = items[3]

m = 93

9 i = 3 + 1

i = 4

6 while 4 < len([-53, 54, 35, 93, -50]): --- True

7 if 93 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-53, 54, 35, 93, -50]): --- False

10 return 93

7. unique([26, 26, -5]) = [26, -5]

1 def unique(items=[26, 26, -5])

2 res = []

3 i = 0

4 while 0 < len([26, 26, -5]): --- True

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

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

res = [26]

7 i = 0 + 1

i = 1

4 while 1 < len([26, 26, -5]): --- True

5 if len([26]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([26, 26, -5]): --- True

5 if len([26]) == 0 or res[-1] != items[2]: --- True

6 res = [26] + [items[2]]

res = [26, -5]

7 i = 2 + 1

i = 3

4 while 3 < len([26, 26, -5]): --- False

8 return [26, -5]

8. join(',', [33, 80, 52, 83]) = '33,80,52,83'

1 def join(sep=,, items=[33, 80, 52, 83])

2 res = ''

3 if len([33, 80, 52, 83]) > 0: --- True

4 res = str(items[0])

res = '33'

5 items = items[1:]

items = [80, 52, 83]

6 while len([80, 52, 83]) > 0: --- True

7 res = '33' + ',' + str(items[0])

res = '33,80'

8 items = items[1:]

items = [52, 83]

6 while len([52, 83]) > 0: --- True

7 res = '33,80' + ',' + str(items[0])

res = '33,80,52'

8 items = items[1:]

items = [83]

6 while len([83]) > 0: --- True

7 res = '33,80,52' + ',' + str(items[0])

res = '33,80,52,83'

8 items = items[1:]

items = []

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

9 return '33,80,52,83'

Вариант: 2-1-15

1. gcd(18, 42) = 6

1 def gcd(x=18, y=42)

2 if 18 < 0: --- False

4 if 42 < 0: --- False

6 if 18 == 0: --- False

8 while 42 != 0: --- True

9 rem = 18 % 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(95, 0) = 95

1 def gcd(x=95, y=0)

2 if 95 < 0: --- False

4 if 0 < 0: --- False

6 if 95 == 0: --- False

8 while 0 != 0: --- False

12 return 95

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(-20, 23, 88) = [2.75, -1.6]

3 def square\_equal(a=-20, b=23, c=88)

4 if -20 != 0: --- True

5 D = 23\*23 - 4\*-20\*88

D = 7569

6 if 7569 > 0: --- True

7 x1 = (-23 - sqrt(7569)) / (2\*-20)

x1 = 2.75

8 x2 = (-23 + sqrt(7569)) / (2\*-20)

x2 = -1.6

9 return [2.75, -1.6]

5. square\_equal(49, 0, 88) = []

3 def square\_equal(a=49, b=0, c=88)

4 if 49 != 0: --- True

5 D = 0\*0 - 4\*49\*88

D = -17248

6 if -17248 > 0: --- False

10 elif -17248 == 0: --- False

12 else:

13 return []

6. findmax([-59, 72, -29, 42]) = 72

1 def findmax(items=[-59, 72, -29, 42])

2 if len([-59, 72, -29, 42]) == 0: --- False

4 m = items[0]

m = -59

5 i = 1

6 while 1 < len([-59, 72, -29, 42]): --- True

7 if -59 < items[1]: --- True

8 m = items[1]

m = 72

9 i = 1 + 1

i = 2

6 while 2 < len([-59, 72, -29, 42]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([-59, 72, -29, 42]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([-59, 72, -29, 42]): --- False

10 return 72

7. unique([29, 13, 91, 91]) = [29, 13, 91]

1 def unique(items=[29, 13, 91, 91])

2 res = []

3 i = 0

4 while 0 < len([29, 13, 91, 91]): --- True

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

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

res = [29]

7 i = 0 + 1

i = 1

4 while 1 < len([29, 13, 91, 91]): --- True

5 if len([29]) == 0 or res[-1] != items[1]: --- True

6 res = [29] + [items[1]]

res = [29, 13]

7 i = 1 + 1

i = 2

4 while 2 < len([29, 13, 91, 91]): --- True

5 if len([29, 13]) == 0 or res[-1] != items[2]: --- True

6 res = [29, 13] + [items[2]]

res = [29, 13, 91]

7 i = 2 + 1

i = 3

4 while 3 < len([29, 13, 91, 91]): --- True

5 if len([29, 13, 91]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([29, 13, 91, 91]): --- False

8 return [29, 13, 91]

8. join(';', [68, 72, 91]) = '68;72;91'

1 def join(sep=;, items=[68, 72, 91])

2 res = ''

3 if len([68, 72, 91]) > 0: --- True

4 res = str(items[0])

res = '68'

5 items = items[1:]

items = [72, 91]

6 while len([72, 91]) > 0: --- True

7 res = '68' + ';' + str(items[0])

res = '68;72'

8 items = items[1:]

items = [91]

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

7 res = '68;72' + ';' + str(items[0])

res = '68;72;91'

8 items = items[1:]

items = []

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

9 return '68;72;91'

Вариант: 2-1-16

1. gcd(8, -100) = 4

1 def gcd(x=8, y=-100)

2 if 8 < 0: --- False

4 if -100 < 0: --- True

5 y = --100

y = 100

6 if 8 == 0: --- False

8 while 100 != 0: --- True

9 rem = 8 % 100

rem = 8

10 x = 100

11 y = 8

8 while 8 != 0: --- True

9 rem = 100 % 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(82, 0) = 82

1 def gcd(x=82, y=0)

2 if 82 < 0: --- False

4 if 0 < 0: --- False

6 if 82 == 0: --- False

8 while 0 != 0: --- False

12 return 82

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(0, -100, -78) = [-0.78]

3 def square\_equal(a=0, b=-100, c=-78)

4 if 0 != 0: --- False

14 else:

15 if -100 != 0: --- True

16 return [-0.78]

5. square\_equal(6, -13, 39) = []

3 def square\_equal(a=6, b=-13, c=39)

4 if 6 != 0: --- True

5 D = -13\*-13 - 4\*6\*39

D = -767

6 if -767 > 0: --- False

10 elif -767 == 0: --- False

12 else:

13 return []

6. findmax([-35, 60, -99, 47, 32, 50]) = 60

1 def findmax(items=[-35, 60, -99, 47, 32, 50])

2 if len([-35, 60, -99, 47, 32, 50]) == 0: --- False

4 m = items[0]

m = -35

5 i = 1

6 while 1 < len([-35, 60, -99, 47, 32, 50]): --- True

7 if -35 < items[1]: --- True

8 m = items[1]

m = 60

9 i = 1 + 1

i = 2

6 while 2 < len([-35, 60, -99, 47, 32, 50]): --- True

7 if 60 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-35, 60, -99, 47, 32, 50]): --- True

7 if 60 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-35, 60, -99, 47, 32, 50]): --- True

7 if 60 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-35, 60, -99, 47, 32, 50]): --- True

7 if 60 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([-35, 60, -99, 47, 32, 50]): --- False

10 return 60

7. unique([-72, -72, 70, -53]) = [-72, 70, -53]

1 def unique(items=[-72, -72, 70, -53])

2 res = []

3 i = 0

4 while 0 < len([-72, -72, 70, -53]): --- True

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

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

res = [-72]

7 i = 0 + 1

i = 1

4 while 1 < len([-72, -72, 70, -53]): --- True

5 if len([-72]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-72, -72, 70, -53]): --- True

5 if len([-72]) == 0 or res[-1] != items[2]: --- True

6 res = [-72] + [items[2]]

res = [-72, 70]

7 i = 2 + 1

i = 3

4 while 3 < len([-72, -72, 70, -53]): --- True

5 if len([-72, 70]) == 0 or res[-1] != items[3]: --- True

6 res = [-72, 70] + [items[3]]

res = [-72, 70, -53]

7 i = 3 + 1

i = 4

4 while 4 < len([-72, -72, 70, -53]): --- False

8 return [-72, 70, -53]

8. join(';', [4, 63, 98]) = '4;63;98'

1 def join(sep=;, items=[4, 63, 98])

2 res = ''

3 if len([4, 63, 98]) > 0: --- True

4 res = str(items[0])

res = '4'

5 items = items[1:]

items = [63, 98]

6 while len([63, 98]) > 0: --- True

7 res = '4' + ';' + str(items[0])

res = '4;63'

8 items = items[1:]

items = [98]

6 while len([98]) > 0: --- True

7 res = '4;63' + ';' + str(items[0])

res = '4;63;98'

8 items = items[1:]

items = []

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

9 return '4;63;98'

Вариант: 2-1-17

1. gcd(-18, 93) = 3

1 def gcd(x=-18, y=93)

2 if -18 < 0: --- True

3 x = --18

x = 18

4 if 93 < 0: --- False

6 if 18 == 0: --- False

8 while 93 != 0: --- True

9 rem = 18 % 93

rem = 18

10 x = 93

11 y = 18

8 while 18 != 0: --- True

9 rem = 93 % 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(20, 0) = 20

1 def gcd(x=20, y=0)

2 if 20 < 0: --- False

4 if 0 < 0: --- False

6 if 20 == 0: --- False

8 while 0 != 0: --- False

12 return 20

3. hex(212) = 'D4'

3 def hex(number=212)

4 if 212 == 0: --- False

6 res = ''

7 while 212 > 0: --- True

8 digit = 212 % 16

digit = 4

9 res = DIGITS[4] + ''

res = '4'

10 number = 212 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + '4'

res = 'D4'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'D4'

4. square\_equal(1, -35, 0) = [0.0, 35.0]

3 def square\_equal(a=1, b=-35, c=0)

4 if 1 != 0: --- True

5 D = -35\*-35 - 4\*1\*0

D = 1225

6 if 1225 > 0: --- True

7 x1 = (--35 - sqrt(1225)) / (2\*1)

x1 = 0.0

8 x2 = (--35 + sqrt(1225)) / (2\*1)

x2 = 35.0

9 return [0.0, 35.0]

5. square\_equal(-32, -2, -35) = []

3 def square\_equal(a=-32, b=-2, c=-35)

4 if -32 != 0: --- True

5 D = -2\*-2 - 4\*-32\*-35

D = -4476

6 if -4476 > 0: --- False

10 elif -4476 == 0: --- False

12 else:

13 return []

6. findmax([-20, -26, -87, -23, -6]) = -6

1 def findmax(items=[-20, -26, -87, -23, -6])

2 if len([-20, -26, -87, -23, -6]) == 0: --- False

4 m = items[0]

m = -20

5 i = 1

6 while 1 < len([-20, -26, -87, -23, -6]): --- True

7 if -20 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([-20, -26, -87, -23, -6]): --- True

7 if -20 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-20, -26, -87, -23, -6]): --- True

7 if -20 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-20, -26, -87, -23, -6]): --- True

7 if -20 < items[4]: --- True

8 m = items[4]

m = -6

9 i = 4 + 1

i = 5

6 while 5 < len([-20, -26, -87, -23, -6]): --- False

10 return -6

7. unique([47, -73, -73, 59]) = [47, -73, 59]

1 def unique(items=[47, -73, -73, 59])

2 res = []

3 i = 0

4 while 0 < len([47, -73, -73, 59]): --- True

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

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

res = [47]

7 i = 0 + 1

i = 1

4 while 1 < len([47, -73, -73, 59]): --- True

5 if len([47]) == 0 or res[-1] != items[1]: --- True

6 res = [47] + [items[1]]

res = [47, -73]

7 i = 1 + 1

i = 2

4 while 2 < len([47, -73, -73, 59]): --- True

5 if len([47, -73]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([47, -73, -73, 59]): --- True

5 if len([47, -73]) == 0 or res[-1] != items[3]: --- True

6 res = [47, -73] + [items[3]]

res = [47, -73, 59]

7 i = 3 + 1

i = 4

4 while 4 < len([47, -73, -73, 59]): --- False

8 return [47, -73, 59]

8. join('+', [42, 68, 67]) = '42+68+67'

1 def join(sep=+, items=[42, 68, 67])

2 res = ''

3 if len([42, 68, 67]) > 0: --- True

4 res = str(items[0])

res = '42'

5 items = items[1:]

items = [68, 67]

6 while len([68, 67]) > 0: --- True

7 res = '42' + '+' + str(items[0])

res = '42+68'

8 items = items[1:]

items = [67]

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

7 res = '42+68' + '+' + str(items[0])

res = '42+68+67'

8 items = items[1:]

items = []

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

9 return '42+68+67'

Вариант: 2-1-18

1. gcd(50, -55) = 5

1 def gcd(x=50, y=-55)

2 if 50 < 0: --- False

4 if -55 < 0: --- True

5 y = --55

y = 55

6 if 50 == 0: --- False

8 while 55 != 0: --- True

9 rem = 50 % 55

rem = 50

10 x = 55

11 y = 50

8 while 50 != 0: --- True

9 rem = 55 % 50

rem = 5

10 x = 50

11 y = 5

8 while 5 != 0: --- True

9 rem = 50 % 5

rem = 0

10 x = 5

11 y = 0

8 while 0 != 0: --- False

12 return 5

2. gcd(0, -18) = 18

1 def gcd(x=0, y=-18)

2 if 0 < 0: --- False

4 if -18 < 0: --- True

5 y = --18

y = 18

6 if 0 == 0: --- True

7 return 18

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(-2, 19, -17) = [8.5, 1.0]

3 def square\_equal(a=-2, b=19, c=-17)

4 if -2 != 0: --- True

5 D = 19\*19 - 4\*-2\*-17

D = 225

6 if 225 > 0: --- True

7 x1 = (-19 - sqrt(225)) / (2\*-2)

x1 = 8.5

8 x2 = (-19 + sqrt(225)) / (2\*-2)

x2 = 1.0

9 return [8.5, 1.0]

5. square\_equal(92, -77, 26) = []

3 def square\_equal(a=92, b=-77, c=26)

4 if 92 != 0: --- True

5 D = -77\*-77 - 4\*92\*26

D = -3639

6 if -3639 > 0: --- False

10 elif -3639 == 0: --- False

12 else:

13 return []

6. findmax([-97, -60, 20, 31]) = 31

1 def findmax(items=[-97, -60, 20, 31])

2 if len([-97, -60, 20, 31]) == 0: --- False

4 m = items[0]

m = -97

5 i = 1

6 while 1 < len([-97, -60, 20, 31]): --- True

7 if -97 < items[1]: --- True

8 m = items[1]

m = -60

9 i = 1 + 1

i = 2

6 while 2 < len([-97, -60, 20, 31]): --- True

7 if -60 < items[2]: --- True

8 m = items[2]

m = 20

9 i = 2 + 1

i = 3

6 while 3 < len([-97, -60, 20, 31]): --- True

7 if 20 < items[3]: --- True

8 m = items[3]

m = 31

9 i = 3 + 1

i = 4

6 while 4 < len([-97, -60, 20, 31]): --- False

10 return 31

7. unique([17, 17, 21, 17]) = [17, 21, 17]

1 def unique(items=[17, 17, 21, 17])

2 res = []

3 i = 0

4 while 0 < len([17, 17, 21, 17]): --- True

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

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

res = [17]

7 i = 0 + 1

i = 1

4 while 1 < len([17, 17, 21, 17]): --- True

5 if len([17]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([17, 17, 21, 17]): --- True

5 if len([17]) == 0 or res[-1] != items[2]: --- True

6 res = [17] + [items[2]]

res = [17, 21]

7 i = 2 + 1

i = 3

4 while 3 < len([17, 17, 21, 17]): --- True

5 if len([17, 21]) == 0 or res[-1] != items[3]: --- True

6 res = [17, 21] + [items[3]]

res = [17, 21, 17]

7 i = 3 + 1

i = 4

4 while 4 < len([17, 17, 21, 17]): --- False

8 return [17, 21, 17]

8. join(';', [21, 28, 13, 28]) = '21;28;13;28'

1 def join(sep=;, items=[21, 28, 13, 28])

2 res = ''

3 if len([21, 28, 13, 28]) > 0: --- True

4 res = str(items[0])

res = '21'

5 items = items[1:]

items = [28, 13, 28]

6 while len([28, 13, 28]) > 0: --- True

7 res = '21' + ';' + str(items[0])

res = '21;28'

8 items = items[1:]

items = [13, 28]

6 while len([13, 28]) > 0: --- True

7 res = '21;28' + ';' + str(items[0])

res = '21;28;13'

8 items = items[1:]

items = [28]

6 while len([28]) > 0: --- True

7 res = '21;28;13' + ';' + str(items[0])

res = '21;28;13;28'

8 items = items[1:]

items = []

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

9 return '21;28;13;28'

Вариант: 2-1-19

1. gcd(-44, -24) = 4

1 def gcd(x=-44, y=-24)

2 if -44 < 0: --- True

3 x = --44

x = 44

4 if -24 < 0: --- True

5 y = --24

y = 24

6 if 44 == 0: --- False

8 while 24 != 0: --- True

9 rem = 44 % 24

rem = 20

10 x = 24

11 y = 20

8 while 20 != 0: --- True

9 rem = 24 % 20

rem = 4

10 x = 20

11 y = 4

8 while 4 != 0: --- True

9 rem = 20 % 4

rem = 0

10 x = 4

11 y = 0

8 while 0 != 0: --- False

12 return 4

2. gcd(-63, 0) = 63

1 def gcd(x=-63, y=0)

2 if -63 < 0: --- True

3 x = --63

x = 63

4 if 0 < 0: --- False

6 if 63 == 0: --- False

8 while 0 != 0: --- False

12 return 63

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(50, -27, 0) = [0.0, 0.54]

3 def square\_equal(a=50, b=-27, c=0)

4 if 50 != 0: --- True

5 D = -27\*-27 - 4\*50\*0

D = 729

6 if 729 > 0: --- True

7 x1 = (--27 - sqrt(729)) / (2\*50)

x1 = 0.0

8 x2 = (--27 + sqrt(729)) / (2\*50)

x2 = 0.54

9 return [0.0, 0.54]

5. square\_equal(60, 94, 41) = []

3 def square\_equal(a=60, b=94, c=41)

4 if 60 != 0: --- True

5 D = 94\*94 - 4\*60\*41

D = -1004

6 if -1004 > 0: --- False

10 elif -1004 == 0: --- False

12 else:

13 return []

6. findmax([93, 14, -33, 59, -84, -20]) = 93

1 def findmax(items=[93, 14, -33, 59, -84, -20])

2 if len([93, 14, -33, 59, -84, -20]) == 0: --- False

4 m = items[0]

m = 93

5 i = 1

6 while 1 < len([93, 14, -33, 59, -84, -20]): --- True

7 if 93 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([93, 14, -33, 59, -84, -20]): --- True

7 if 93 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([93, 14, -33, 59, -84, -20]): --- True

7 if 93 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([93, 14, -33, 59, -84, -20]): --- True

7 if 93 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([93, 14, -33, 59, -84, -20]): --- True

7 if 93 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([93, 14, -33, 59, -84, -20]): --- False

10 return 93

7. unique([-99, -16, -16, 45]) = [-99, -16, 45]

1 def unique(items=[-99, -16, -16, 45])

2 res = []

3 i = 0

4 while 0 < len([-99, -16, -16, 45]): --- True

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

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

res = [-99]

7 i = 0 + 1

i = 1

4 while 1 < len([-99, -16, -16, 45]): --- True

5 if len([-99]) == 0 or res[-1] != items[1]: --- True

6 res = [-99] + [items[1]]

res = [-99, -16]

7 i = 1 + 1

i = 2

4 while 2 < len([-99, -16, -16, 45]): --- True

5 if len([-99, -16]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-99, -16, -16, 45]): --- True

5 if len([-99, -16]) == 0 or res[-1] != items[3]: --- True

6 res = [-99, -16] + [items[3]]

res = [-99, -16, 45]

7 i = 3 + 1

i = 4

4 while 4 < len([-99, -16, -16, 45]): --- False

8 return [-99, -16, 45]

8. join(';', [8, 13, 72]) = '8;13;72'

1 def join(sep=;, items=[8, 13, 72])

2 res = ''

3 if len([8, 13, 72]) > 0: --- True

4 res = str(items[0])

res = '8'

5 items = items[1:]

items = [13, 72]

6 while len([13, 72]) > 0: --- True

7 res = '8' + ';' + str(items[0])

res = '8;13'

8 items = items[1:]

items = [72]

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

7 res = '8;13' + ';' + str(items[0])

res = '8;13;72'

8 items = items[1:]

items = []

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

9 return '8;13;72'

Вариант: 2-1-20

1. gcd(-87, 27) = 3

1 def gcd(x=-87, y=27)

2 if -87 < 0: --- True

3 x = --87

x = 87

4 if 27 < 0: --- False

6 if 87 == 0: --- False

8 while 27 != 0: --- True

9 rem = 87 % 27

rem = 6

10 x = 27

11 y = 6

8 while 6 != 0: --- True

9 rem = 27 % 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(0, 34) = 34

1 def gcd(x=0, y=34)

2 if 0 < 0: --- False

4 if 34 < 0: --- False

6 if 0 == 0: --- True

7 return 34

3. hex(255) = 'FF'

3 def hex(number=255)

4 if 255 == 0: --- False

6 res = ''

7 while 255 > 0: --- True

8 digit = 255 % 16

digit = 15

9 res = DIGITS[15] + ''

res = 'F'

10 number = 255 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + 'F'

res = 'FF'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'FF'

4. square\_equal(4, 38, 60) = [-7.5, -2.0]

3 def square\_equal(a=4, b=38, c=60)

4 if 4 != 0: --- True

5 D = 38\*38 - 4\*4\*60

D = 484

6 if 484 > 0: --- True

7 x1 = (-38 - sqrt(484)) / (2\*4)

x1 = -7.5

8 x2 = (-38 + sqrt(484)) / (2\*4)

x2 = -2.0

9 return [-7.5, -2.0]

5. square\_equal(76, 66, 22) = []

3 def square\_equal(a=76, b=66, c=22)

4 if 76 != 0: --- True

5 D = 66\*66 - 4\*76\*22

D = -2332

6 if -2332 > 0: --- False

10 elif -2332 == 0: --- False

12 else:

13 return []

6. findmax([-24, -63, 59, -68]) = 59

1 def findmax(items=[-24, -63, 59, -68])

2 if len([-24, -63, 59, -68]) == 0: --- False

4 m = items[0]

m = -24

5 i = 1

6 while 1 < len([-24, -63, 59, -68]): --- True

7 if -24 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([-24, -63, 59, -68]): --- True

7 if -24 < items[2]: --- True

8 m = items[2]

m = 59

9 i = 2 + 1

i = 3

6 while 3 < len([-24, -63, 59, -68]): --- True

7 if 59 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-24, -63, 59, -68]): --- False

10 return 59

7. unique([36, 36, 6]) = [36, 6]

1 def unique(items=[36, 36, 6])

2 res = []

3 i = 0

4 while 0 < len([36, 36, 6]): --- True

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

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

res = [36]

7 i = 0 + 1

i = 1

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

5 if len([36]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

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

5 if len([36]) == 0 or res[-1] != items[2]: --- True

6 res = [36] + [items[2]]

res = [36, 6]

7 i = 2 + 1

i = 3

4 while 3 < len([36, 36, 6]): --- False

8 return [36, 6]

8. join(',', [88, 8, 27, 39]) = '88,8,27,39'

1 def join(sep=,, items=[88, 8, 27, 39])

2 res = ''

3 if len([88, 8, 27, 39]) > 0: --- True

4 res = str(items[0])

res = '88'

5 items = items[1:]

items = [8, 27, 39]

6 while len([8, 27, 39]) > 0: --- True

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

res = '88,8'

8 items = items[1:]

items = [27, 39]

6 while len([27, 39]) > 0: --- True

7 res = '88,8' + ',' + str(items[0])

res = '88,8,27'

8 items = items[1:]

items = [39]

6 while len([39]) > 0: --- True

7 res = '88,8,27' + ',' + str(items[0])

res = '88,8,27,39'

8 items = items[1:]

items = []

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

9 return '88,8,27,39'

Вариант: 2-1-21

1. gcd(32, 20) = 4

1 def gcd(x=32, y=20)

2 if 32 < 0: --- False

4 if 20 < 0: --- False

6 if 32 == 0: --- False

8 while 20 != 0: --- True

9 rem = 32 % 20

rem = 12

10 x = 20

11 y = 12

8 while 12 != 0: --- True

9 rem = 20 % 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, 28) = 28

1 def gcd(x=0, y=28)

2 if 0 < 0: --- False

4 if 28 < 0: --- False

6 if 0 == 0: --- True

7 return 28

3. hex(218) = 'DA'

3 def hex(number=218)

4 if 218 == 0: --- False

6 res = ''

7 while 218 > 0: --- True

8 digit = 218 % 16

digit = 10

9 res = DIGITS[10] + ''

res = 'A'

10 number = 218 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + 'A'

res = 'DA'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'DA'

4. square\_equal(2, 59, 57) = [-28.5, -1.0]

3 def square\_equal(a=2, b=59, c=57)

4 if 2 != 0: --- True

5 D = 59\*59 - 4\*2\*57

D = 3025

6 if 3025 > 0: --- True

7 x1 = (-59 - sqrt(3025)) / (2\*2)

x1 = -28.5

8 x2 = (-59 + sqrt(3025)) / (2\*2)

x2 = -1.0

9 return [-28.5, -1.0]

5. square\_equal(-53, 19, -22) = []

3 def square\_equal(a=-53, b=19, c=-22)

4 if -53 != 0: --- True

5 D = 19\*19 - 4\*-53\*-22

D = -4303

6 if -4303 > 0: --- False

10 elif -4303 == 0: --- False

12 else:

13 return []

6. findmax([-84, 92, 95, -13]) = 95

1 def findmax(items=[-84, 92, 95, -13])

2 if len([-84, 92, 95, -13]) == 0: --- False

4 m = items[0]

m = -84

5 i = 1

6 while 1 < len([-84, 92, 95, -13]): --- True

7 if -84 < items[1]: --- True

8 m = items[1]

m = 92

9 i = 1 + 1

i = 2

6 while 2 < len([-84, 92, 95, -13]): --- True

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

8 m = items[2]

m = 95

9 i = 2 + 1

i = 3

6 while 3 < len([-84, 92, 95, -13]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([-84, 92, 95, -13]): --- False

10 return 95

7. unique([-38, 93, 93, -4]) = [-38, 93, -4]

1 def unique(items=[-38, 93, 93, -4])

2 res = []

3 i = 0

4 while 0 < len([-38, 93, 93, -4]): --- True

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

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

res = [-38]

7 i = 0 + 1

i = 1

4 while 1 < len([-38, 93, 93, -4]): --- True

5 if len([-38]) == 0 or res[-1] != items[1]: --- True

6 res = [-38] + [items[1]]

res = [-38, 93]

7 i = 1 + 1

i = 2

4 while 2 < len([-38, 93, 93, -4]): --- True

5 if len([-38, 93]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-38, 93, 93, -4]): --- True

5 if len([-38, 93]) == 0 or res[-1] != items[3]: --- True

6 res = [-38, 93] + [items[3]]

res = [-38, 93, -4]

7 i = 3 + 1

i = 4

4 while 4 < len([-38, 93, 93, -4]): --- False

8 return [-38, 93, -4]

8. join(',', [44, 49, 85]) = '44,49,85'

1 def join(sep=,, items=[44, 49, 85])

2 res = ''

3 if len([44, 49, 85]) > 0: --- True

4 res = str(items[0])

res = '44'

5 items = items[1:]

items = [49, 85]

6 while len([49, 85]) > 0: --- True

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

res = '44,49'

8 items = items[1:]

items = [85]

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

7 res = '44,49' + ',' + str(items[0])

res = '44,49,85'

8 items = items[1:]

items = []

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

9 return '44,49,85'

Вариант: 2-1-22

1. gcd(70, -49) = 7

1 def gcd(x=70, y=-49)

2 if 70 < 0: --- False

4 if -49 < 0: --- True

5 y = --49

y = 49

6 if 70 == 0: --- False

8 while 49 != 0: --- True

9 rem = 70 % 49

rem = 21

10 x = 49

11 y = 21

8 while 21 != 0: --- True

9 rem = 49 % 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(-32, 0) = 32

1 def gcd(x=-32, y=0)

2 if -32 < 0: --- True

3 x = --32

x = 32

4 if 0 < 0: --- False

6 if 32 == 0: --- False

8 while 0 != 0: --- False

12 return 32

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(12, -36, 27) = [1.5]

3 def square\_equal(a=12, b=-36, c=27)

4 if 12 != 0: --- True

5 D = -36\*-36 - 4\*12\*27

D = 0

6 if 0 > 0: --- False

10 elif 0 == 0: --- True

11 return [1.5]

5. square\_equal(45, 60, 77) = []

3 def square\_equal(a=45, b=60, c=77)

4 if 45 != 0: --- True

5 D = 60\*60 - 4\*45\*77

D = -10260

6 if -10260 > 0: --- False

10 elif -10260 == 0: --- False

12 else:

13 return []

6. findmax([40, 92, 80, 86, -31]) = 92

1 def findmax(items=[40, 92, 80, 86, -31])

2 if len([40, 92, 80, 86, -31]) == 0: --- False

4 m = items[0]

m = 40

5 i = 1

6 while 1 < len([40, 92, 80, 86, -31]): --- True

7 if 40 < items[1]: --- True

8 m = items[1]

m = 92

9 i = 1 + 1

i = 2

6 while 2 < len([40, 92, 80, 86, -31]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([40, 92, 80, 86, -31]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([40, 92, 80, 86, -31]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([40, 92, 80, 86, -31]): --- False

10 return 92

7. unique([11, 11, 72, 97]) = [11, 72, 97]

1 def unique(items=[11, 11, 72, 97])

2 res = []

3 i = 0

4 while 0 < len([11, 11, 72, 97]): --- True

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

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

res = [11]

7 i = 0 + 1

i = 1

4 while 1 < len([11, 11, 72, 97]): --- True

5 if len([11]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([11, 11, 72, 97]): --- True

5 if len([11]) == 0 or res[-1] != items[2]: --- True

6 res = [11] + [items[2]]

res = [11, 72]

7 i = 2 + 1

i = 3

4 while 3 < len([11, 11, 72, 97]): --- True

5 if len([11, 72]) == 0 or res[-1] != items[3]: --- True

6 res = [11, 72] + [items[3]]

res = [11, 72, 97]

7 i = 3 + 1

i = 4

4 while 4 < len([11, 11, 72, 97]): --- False

8 return [11, 72, 97]

8. join('+', [49, 23, 78]) = '49+23+78'

1 def join(sep=+, items=[49, 23, 78])

2 res = ''

3 if len([49, 23, 78]) > 0: --- True

4 res = str(items[0])

res = '49'

5 items = items[1:]

items = [23, 78]

6 while len([23, 78]) > 0: --- True

7 res = '49' + '+' + str(items[0])

res = '49+23'

8 items = items[1:]

items = [78]

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

7 res = '49+23' + '+' + str(items[0])

res = '49+23+78'

8 items = items[1:]

items = []

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

9 return '49+23+78'

Вариант: 2-1-23

1. gcd(-42, 96) = 6

1 def gcd(x=-42, y=96)

2 if -42 < 0: --- True

3 x = --42

x = 42

4 if 96 < 0: --- False

6 if 42 == 0: --- False

8 while 96 != 0: --- True

9 rem = 42 % 96

rem = 42

10 x = 96

11 y = 42

8 while 42 != 0: --- True

9 rem = 96 % 42

rem = 12

10 x = 42

11 y = 12

8 while 12 != 0: --- True

9 rem = 42 % 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, -58) = 58

1 def gcd(x=0, y=-58)

2 if 0 < 0: --- False

4 if -58 < 0: --- True

5 y = --58

y = 58

6 if 0 == 0: --- True

7 return 58

3. hex(186) = 'BA'

3 def hex(number=186)

4 if 186 == 0: --- False

6 res = ''

7 while 186 > 0: --- True

8 digit = 186 % 16

digit = 10

9 res = DIGITS[10] + ''

res = 'A'

10 number = 186 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + 'A'

res = 'BA'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'BA'

4. square\_equal(-10, 49, -22) = [4.4, 0.5]

3 def square\_equal(a=-10, b=49, c=-22)

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

5 D = 49\*49 - 4\*-10\*-22

D = 1521

6 if 1521 > 0: --- True

7 x1 = (-49 - sqrt(1521)) / (2\*-10)

x1 = 4.4

8 x2 = (-49 + sqrt(1521)) / (2\*-10)

x2 = 0.5

9 return [4.4, 0.5]

5. square\_equal(-64, 63, -41) = []

3 def square\_equal(a=-64, b=63, c=-41)

4 if -64 != 0: --- True

5 D = 63\*63 - 4\*-64\*-41

D = -6527

6 if -6527 > 0: --- False

10 elif -6527 == 0: --- False

12 else:

13 return []

6. findmax([96, -63, 55, -40, -80]) = 96

1 def findmax(items=[96, -63, 55, -40, -80])

2 if len([96, -63, 55, -40, -80]) == 0: --- False

4 m = items[0]

m = 96

5 i = 1

6 while 1 < len([96, -63, 55, -40, -80]): --- True

7 if 96 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([96, -63, 55, -40, -80]): --- True

7 if 96 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([96, -63, 55, -40, -80]): --- True

7 if 96 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([96, -63, 55, -40, -80]): --- True

7 if 96 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([96, -63, 55, -40, -80]): --- False

10 return 96

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

1 def unique(items=[-7, 91, 91])

2 res = []

3 i = 0

4 while 0 < len([-7, 91, 91]): --- 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, 91, 91]): --- True

5 if len([-7]) == 0 or res[-1] != items[1]: --- True

6 res = [-7] + [items[1]]

res = [-7, 91]

7 i = 1 + 1

i = 2

4 while 2 < len([-7, 91, 91]): --- True

5 if len([-7, 91]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-7, 91, 91]): --- False

8 return [-7, 91]

8. join(':', [2, 55, 18]) = '2:55:18'

1 def join(sep=:, items=[2, 55, 18])

2 res = ''

3 if len([2, 55, 18]) > 0: --- True

4 res = str(items[0])

res = '2'

5 items = items[1:]

items = [55, 18]

6 while len([55, 18]) > 0: --- True

7 res = '2' + ':' + str(items[0])

res = '2:55'

8 items = items[1:]

items = [18]

6 while len([18]) > 0: --- True

7 res = '2:55' + ':' + str(items[0])

res = '2:55:18'

8 items = items[1:]

items = []

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

9 return '2:55:18'

Вариант: 2-1-24

1. gcd(-84, 98) = 14

1 def gcd(x=-84, y=98)

2 if -84 < 0: --- True

3 x = --84

x = 84

4 if 98 < 0: --- False

6 if 84 == 0: --- False

8 while 98 != 0: --- True

9 rem = 84 % 98

rem = 84

10 x = 98

11 y = 84

8 while 84 != 0: --- True

9 rem = 98 % 84

rem = 14

10 x = 84

11 y = 14

8 while 14 != 0: --- True

9 rem = 84 % 14

rem = 0

10 x = 14

11 y = 0

8 while 0 != 0: --- False

12 return 14

2. gcd(-98, 0) = 98

1 def gcd(x=-98, y=0)

2 if -98 < 0: --- True

3 x = --98

x = 98

4 if 0 < 0: --- False

6 if 98 == 0: --- False

8 while 0 != 0: --- False

12 return 98

3. hex(190) = 'BE'

3 def hex(number=190)

4 if 190 == 0: --- False

6 res = ''

7 while 190 > 0: --- True

8 digit = 190 % 16

digit = 14

9 res = DIGITS[14] + ''

res = 'E'

10 number = 190 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + 'E'

res = 'BE'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'BE'

4. square\_equal(0, 20, 85) = [-4.25]

3 def square\_equal(a=0, b=20, c=85)

4 if 0 != 0: --- False

14 else:

15 if 20 != 0: --- True

16 return [-4.25]

5. square\_equal(-27, -3, -78) = []

3 def square\_equal(a=-27, b=-3, c=-78)

4 if -27 != 0: --- True

5 D = -3\*-3 - 4\*-27\*-78

D = -8415

6 if -8415 > 0: --- False

10 elif -8415 == 0: --- False

12 else:

13 return []

6. findmax([-87, 20, 23, -79]) = 23

1 def findmax(items=[-87, 20, 23, -79])

2 if len([-87, 20, 23, -79]) == 0: --- False

4 m = items[0]

m = -87

5 i = 1

6 while 1 < len([-87, 20, 23, -79]): --- True

7 if -87 < items[1]: --- True

8 m = items[1]

m = 20

9 i = 1 + 1

i = 2

6 while 2 < len([-87, 20, 23, -79]): --- True

7 if 20 < items[2]: --- True

8 m = items[2]

m = 23

9 i = 2 + 1

i = 3

6 while 3 < len([-87, 20, 23, -79]): --- True

7 if 23 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-87, 20, 23, -79]): --- False

10 return 23

7. unique([71, 69, 47, 47]) = [71, 69, 47]

1 def unique(items=[71, 69, 47, 47])

2 res = []

3 i = 0

4 while 0 < len([71, 69, 47, 47]): --- True

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

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

res = [71]

7 i = 0 + 1

i = 1

4 while 1 < len([71, 69, 47, 47]): --- True

5 if len([71]) == 0 or res[-1] != items[1]: --- True

6 res = [71] + [items[1]]

res = [71, 69]

7 i = 1 + 1

i = 2

4 while 2 < len([71, 69, 47, 47]): --- True

5 if len([71, 69]) == 0 or res[-1] != items[2]: --- True

6 res = [71, 69] + [items[2]]

res = [71, 69, 47]

7 i = 2 + 1

i = 3

4 while 3 < len([71, 69, 47, 47]): --- True

5 if len([71, 69, 47]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([71, 69, 47, 47]): --- False

8 return [71, 69, 47]

8. join('+', [92, 62, 55, 44]) = '92+62+55+44'

1 def join(sep=+, items=[92, 62, 55, 44])

2 res = ''

3 if len([92, 62, 55, 44]) > 0: --- True

4 res = str(items[0])

res = '92'

5 items = items[1:]

items = [62, 55, 44]

6 while len([62, 55, 44]) > 0: --- True

7 res = '92' + '+' + str(items[0])

res = '92+62'

8 items = items[1:]

items = [55, 44]

6 while len([55, 44]) > 0: --- True

7 res = '92+62' + '+' + str(items[0])

res = '92+62+55'

8 items = items[1:]

items = [44]

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

7 res = '92+62+55' + '+' + str(items[0])

res = '92+62+55+44'

8 items = items[1:]

items = []

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

9 return '92+62+55+44'

Вариант: 2-1-25

1. gcd(-75, 51) = 3

1 def gcd(x=-75, y=51)

2 if -75 < 0: --- True

3 x = --75

x = 75

4 if 51 < 0: --- False

6 if 75 == 0: --- False

8 while 51 != 0: --- True

9 rem = 75 % 51

rem = 24

10 x = 51

11 y = 24

8 while 24 != 0: --- True

9 rem = 51 % 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, 41) = 41

1 def gcd(x=0, y=41)

2 if 0 < 0: --- False

4 if 41 < 0: --- False

6 if 0 == 0: --- True

7 return 41

3. hex(164) = 'A4'

3 def hex(number=164)

4 if 164 == 0: --- False

6 res = ''

7 while 164 > 0: --- True

8 digit = 164 % 16

digit = 4

9 res = DIGITS[4] + ''

res = '4'

10 number = 164 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + '4'

res = 'A4'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'A4'

4. square\_equal(-8, 68, 0) = [8.5, -0.0]

3 def square\_equal(a=-8, b=68, c=0)

4 if -8 != 0: --- True

5 D = 68\*68 - 4\*-8\*0

D = 4624

6 if 4624 > 0: --- True

7 x1 = (-68 - sqrt(4624)) / (2\*-8)

x1 = 8.5

8 x2 = (-68 + sqrt(4624)) / (2\*-8)

x2 = -0.0

9 return [8.5, -0.0]

5. square\_equal(54, 54, 81) = []

3 def square\_equal(a=54, b=54, c=81)

4 if 54 != 0: --- True

5 D = 54\*54 - 4\*54\*81

D = -14580

6 if -14580 > 0: --- False

10 elif -14580 == 0: --- False

12 else:

13 return []

6. findmax([57, 72, 48, -42, -35, -88]) = 72

1 def findmax(items=[57, 72, 48, -42, -35, -88])

2 if len([57, 72, 48, -42, -35, -88]) == 0: --- False

4 m = items[0]

m = 57

5 i = 1

6 while 1 < len([57, 72, 48, -42, -35, -88]): --- True

7 if 57 < items[1]: --- True

8 m = items[1]

m = 72

9 i = 1 + 1

i = 2

6 while 2 < len([57, 72, 48, -42, -35, -88]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([57, 72, 48, -42, -35, -88]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([57, 72, 48, -42, -35, -88]): --- True

7 if 72 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([57, 72, 48, -42, -35, -88]): --- True

7 if 72 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([57, 72, 48, -42, -35, -88]): --- False

10 return 72

7. unique([-90, -92, -92]) = [-90, -92]

1 def unique(items=[-90, -92, -92])

2 res = []

3 i = 0

4 while 0 < len([-90, -92, -92]): --- True

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

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

res = [-90]

7 i = 0 + 1

i = 1

4 while 1 < len([-90, -92, -92]): --- True

5 if len([-90]) == 0 or res[-1] != items[1]: --- True

6 res = [-90] + [items[1]]

res = [-90, -92]

7 i = 1 + 1

i = 2

4 while 2 < len([-90, -92, -92]): --- True

5 if len([-90, -92]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-90, -92, -92]): --- False

8 return [-90, -92]

8. join(',', [27, 11, 74]) = '27,11,74'

1 def join(sep=,, items=[27, 11, 74])

2 res = ''

3 if len([27, 11, 74]) > 0: --- True

4 res = str(items[0])

res = '27'

5 items = items[1:]

items = [11, 74]

6 while len([11, 74]) > 0: --- True

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

res = '27,11'

8 items = items[1:]

items = [74]

6 while len([74]) > 0: --- True

7 res = '27,11' + ',' + str(items[0])

res = '27,11,74'

8 items = items[1:]

items = []

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

9 return '27,11,74'

Вариант: 2-1-26

1. gcd(-91, -28) = 7

1 def gcd(x=-91, y=-28)

2 if -91 < 0: --- True

3 x = --91

x = 91

4 if -28 < 0: --- True

5 y = --28

y = 28

6 if 91 == 0: --- False

8 while 28 != 0: --- True

9 rem = 91 % 28

rem = 7

10 x = 28

11 y = 7

8 while 7 != 0: --- True

9 rem = 28 % 7

rem = 0

10 x = 7

11 y = 0

8 while 0 != 0: --- False

12 return 7

2. gcd(57, 0) = 57

1 def gcd(x=57, y=0)

2 if 57 < 0: --- False

4 if 0 < 0: --- False

6 if 57 == 0: --- False

8 while 0 != 0: --- False

12 return 57

3. hex(228) = 'E4'

3 def hex(number=228)

4 if 228 == 0: --- False

6 res = ''

7 while 228 > 0: --- True

8 digit = 228 % 16

digit = 4

9 res = DIGITS[4] + ''

res = '4'

10 number = 228 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + '4'

res = 'E4'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'E4'

4. square\_equal(100, 71, 0) = [-0.71, 0.0]

3 def square\_equal(a=100, b=71, c=0)

4 if 100 != 0: --- True

5 D = 71\*71 - 4\*100\*0

D = 5041

6 if 5041 > 0: --- True

7 x1 = (-71 - sqrt(5041)) / (2\*100)

x1 = -0.71

8 x2 = (-71 + sqrt(5041)) / (2\*100)

x2 = 0.0

9 return [-0.71, 0.0]

5. square\_equal(78, -10, 40) = []

3 def square\_equal(a=78, b=-10, c=40)

4 if 78 != 0: --- True

5 D = -10\*-10 - 4\*78\*40

D = -12380

6 if -12380 > 0: --- False

10 elif -12380 == 0: --- False

12 else:

13 return []

6. findmax([99, 35, -15, -4, -9]) = 99

1 def findmax(items=[99, 35, -15, -4, -9])

2 if len([99, 35, -15, -4, -9]) == 0: --- False

4 m = items[0]

m = 99

5 i = 1

6 while 1 < len([99, 35, -15, -4, -9]): --- True

7 if 99 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([99, 35, -15, -4, -9]): --- True

7 if 99 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([99, 35, -15, -4, -9]): --- True

7 if 99 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([99, 35, -15, -4, -9]): --- True

7 if 99 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([99, 35, -15, -4, -9]): --- False

10 return 99

7. unique([10, 70, 70, -58]) = [10, 70, -58]

1 def unique(items=[10, 70, 70, -58])

2 res = []

3 i = 0

4 while 0 < len([10, 70, 70, -58]): --- True

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

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

res = [10]

7 i = 0 + 1

i = 1

4 while 1 < len([10, 70, 70, -58]): --- True

5 if len([10]) == 0 or res[-1] != items[1]: --- True

6 res = [10] + [items[1]]

res = [10, 70]

7 i = 1 + 1

i = 2

4 while 2 < len([10, 70, 70, -58]): --- True

5 if len([10, 70]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([10, 70, 70, -58]): --- True

5 if len([10, 70]) == 0 or res[-1] != items[3]: --- True

6 res = [10, 70] + [items[3]]

res = [10, 70, -58]

7 i = 3 + 1

i = 4

4 while 4 < len([10, 70, 70, -58]): --- False

8 return [10, 70, -58]

8. join(':', [3, 62, 2, 40]) = '3:62:2:40'

1 def join(sep=:, items=[3, 62, 2, 40])

2 res = ''

3 if len([3, 62, 2, 40]) > 0: --- True

4 res = str(items[0])

res = '3'

5 items = items[1:]

items = [62, 2, 40]

6 while len([62, 2, 40]) > 0: --- True

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

res = '3:62'

8 items = items[1:]

items = [2, 40]

6 while len([2, 40]) > 0: --- True

7 res = '3:62' + ':' + str(items[0])

res = '3:62:2'

8 items = items[1:]

items = [40]

6 while len([40]) > 0: --- True

7 res = '3:62:2' + ':' + str(items[0])

res = '3:62:2:40'

8 items = items[1:]

items = []

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

9 return '3:62:2:40'

Вариант: 2-1-27

1. gcd(-75, -66) = 3

1 def gcd(x=-75, y=-66)

2 if -75 < 0: --- True

3 x = --75

x = 75

4 if -66 < 0: --- True

5 y = --66

y = 66

6 if 75 == 0: --- False

8 while 66 != 0: --- True

9 rem = 75 % 66

rem = 9

10 x = 66

11 y = 9

8 while 9 != 0: --- True

9 rem = 66 % 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(88, 0) = 88

1 def gcd(x=88, y=0)

2 if 88 < 0: --- False

4 if 0 < 0: --- False

6 if 88 == 0: --- False

8 while 0 != 0: --- False

12 return 88

3. hex(161) = 'A1'

3 def hex(number=161)

4 if 161 == 0: --- False

6 res = ''

7 while 161 > 0: --- True

8 digit = 161 % 16

digit = 1

9 res = DIGITS[1] + ''

res = '1'

10 number = 161 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + '1'

res = 'A1'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'A1'

4. square\_equal(80, 72, 9) = [-0.75, -0.15]

3 def square\_equal(a=80, b=72, c=9)

4 if 80 != 0: --- True

5 D = 72\*72 - 4\*80\*9

D = 2304

6 if 2304 > 0: --- True

7 x1 = (-72 - sqrt(2304)) / (2\*80)

x1 = -0.75

8 x2 = (-72 + sqrt(2304)) / (2\*80)

x2 = -0.15

9 return [-0.75, -0.15]

5. square\_equal(84, 54, 64) = []

3 def square\_equal(a=84, b=54, c=64)

4 if 84 != 0: --- True

5 D = 54\*54 - 4\*84\*64

D = -18588

6 if -18588 > 0: --- False

10 elif -18588 == 0: --- False

12 else:

13 return []

6. findmax([44, 10, -43, -42, -48, 52]) = 52

1 def findmax(items=[44, 10, -43, -42, -48, 52])

2 if len([44, 10, -43, -42, -48, 52]) == 0: --- False

4 m = items[0]

m = 44

5 i = 1

6 while 1 < len([44, 10, -43, -42, -48, 52]): --- True

7 if 44 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([44, 10, -43, -42, -48, 52]): --- True

7 if 44 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([44, 10, -43, -42, -48, 52]): --- True

7 if 44 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([44, 10, -43, -42, -48, 52]): --- True

7 if 44 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([44, 10, -43, -42, -48, 52]): --- True

7 if 44 < items[5]: --- True

8 m = items[5]

m = 52

9 i = 5 + 1

i = 6

6 while 6 < len([44, 10, -43, -42, -48, 52]): --- False

10 return 52

7. unique([64, -74, -74, -84]) = [64, -74, -84]

1 def unique(items=[64, -74, -74, -84])

2 res = []

3 i = 0

4 while 0 < len([64, -74, -74, -84]): --- 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, -74, -74, -84]): --- True

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

6 res = [64] + [items[1]]

res = [64, -74]

7 i = 1 + 1

i = 2

4 while 2 < len([64, -74, -74, -84]): --- True

5 if len([64, -74]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([64, -74, -74, -84]): --- True

5 if len([64, -74]) == 0 or res[-1] != items[3]: --- True

6 res = [64, -74] + [items[3]]

res = [64, -74, -84]

7 i = 3 + 1

i = 4

4 while 4 < len([64, -74, -74, -84]): --- False

8 return [64, -74, -84]

8. join(':', [7, 31, 67]) = '7:31:67'

1 def join(sep=:, items=[7, 31, 67])

2 res = ''

3 if len([7, 31, 67]) > 0: --- True

4 res = str(items[0])

res = '7'

5 items = items[1:]

items = [31, 67]

6 while len([31, 67]) > 0: --- True

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

res = '7:31'

8 items = items[1:]

items = [67]

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

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

res = '7:31:67'

8 items = items[1:]

items = []

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

9 return '7:31:67'

Вариант: 2-1-28

1. gcd(-12, 51) = 3

1 def gcd(x=-12, y=51)

2 if -12 < 0: --- True

3 x = --12

x = 12

4 if 51 < 0: --- False

6 if 12 == 0: --- False

8 while 51 != 0: --- True

9 rem = 12 % 51

rem = 12

10 x = 51

11 y = 12

8 while 12 != 0: --- True

9 rem = 51 % 12

rem = 3

10 x = 12

11 y = 3

8 while 3 != 0: --- True

9 rem = 12 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(34, 0) = 34

1 def gcd(x=34, y=0)

2 if 34 < 0: --- False

4 if 0 < 0: --- False

6 if 34 == 0: --- False

8 while 0 != 0: --- False

12 return 34

3. hex(202) = 'CA'

3 def hex(number=202)

4 if 202 == 0: --- False

6 res = ''

7 while 202 > 0: --- True

8 digit = 202 % 16

digit = 10

9 res = DIGITS[10] + ''

res = 'A'

10 number = 202 // 16

number = 12

7 while 12 > 0: --- True

8 digit = 12 % 16

digit = 12

9 res = DIGITS[12] + 'A'

res = 'CA'

10 number = 12 // 16

number = 0

7 while 0 > 0: --- False

11 return 'CA'

4. square\_equal(6, 93, 0) = [-15.5, 0.0]

3 def square\_equal(a=6, b=93, c=0)

4 if 6 != 0: --- True

5 D = 93\*93 - 4\*6\*0

D = 8649

6 if 8649 > 0: --- True

7 x1 = (-93 - sqrt(8649)) / (2\*6)

x1 = -15.5

8 x2 = (-93 + sqrt(8649)) / (2\*6)

x2 = 0.0

9 return [-15.5, 0.0]

5. square\_equal(-100, -75, -84) = []

3 def square\_equal(a=-100, b=-75, c=-84)

4 if -100 != 0: --- True

5 D = -75\*-75 - 4\*-100\*-84

D = -27975

6 if -27975 > 0: --- False

10 elif -27975 == 0: --- False

12 else:

13 return []

6. findmax([9, 75, 31, -75]) = 75

1 def findmax(items=[9, 75, 31, -75])

2 if len([9, 75, 31, -75]) == 0: --- False

4 m = items[0]

m = 9

5 i = 1

6 while 1 < len([9, 75, 31, -75]): --- True

7 if 9 < items[1]: --- True

8 m = items[1]

m = 75

9 i = 1 + 1

i = 2

6 while 2 < len([9, 75, 31, -75]): --- True

7 if 75 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([9, 75, 31, -75]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([9, 75, 31, -75]): --- False

10 return 75

7. unique([-23, -87, -87, -56]) = [-23, -87, -56]

1 def unique(items=[-23, -87, -87, -56])

2 res = []

3 i = 0

4 while 0 < len([-23, -87, -87, -56]): --- True

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

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

res = [-23]

7 i = 0 + 1

i = 1

4 while 1 < len([-23, -87, -87, -56]): --- True

5 if len([-23]) == 0 or res[-1] != items[1]: --- True

6 res = [-23] + [items[1]]

res = [-23, -87]

7 i = 1 + 1

i = 2

4 while 2 < len([-23, -87, -87, -56]): --- True

5 if len([-23, -87]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-23, -87, -87, -56]): --- True

5 if len([-23, -87]) == 0 or res[-1] != items[3]: --- True

6 res = [-23, -87] + [items[3]]

res = [-23, -87, -56]

7 i = 3 + 1

i = 4

4 while 4 < len([-23, -87, -87, -56]): --- False

8 return [-23, -87, -56]

8. join('+', [89, 60, 20, 36]) = '89+60+20+36'

1 def join(sep=+, items=[89, 60, 20, 36])

2 res = ''

3 if len([89, 60, 20, 36]) > 0: --- True

4 res = str(items[0])

res = '89'

5 items = items[1:]

items = [60, 20, 36]

6 while len([60, 20, 36]) > 0: --- True

7 res = '89' + '+' + str(items[0])

res = '89+60'

8 items = items[1:]

items = [20, 36]

6 while len([20, 36]) > 0: --- True

7 res = '89+60' + '+' + str(items[0])

res = '89+60+20'

8 items = items[1:]

items = [36]

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

7 res = '89+60+20' + '+' + str(items[0])

res = '89+60+20+36'

8 items = items[1:]

items = []

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

9 return '89+60+20+36'

Вариант: 2-1-29

1. gcd(64, 68) = 4

1 def gcd(x=64, y=68)

2 if 64 < 0: --- False

4 if 68 < 0: --- False

6 if 64 == 0: --- False

8 while 68 != 0: --- True

9 rem = 64 % 68

rem = 64

10 x = 68

11 y = 64

8 while 64 != 0: --- True

9 rem = 68 % 64

rem = 4

10 x = 64

11 y = 4

8 while 4 != 0: --- True

9 rem = 64 % 4

rem = 0

10 x = 4

11 y = 0

8 while 0 != 0: --- False

12 return 4

2. gcd(0, -10) = 10

1 def gcd(x=0, y=-10)

2 if 0 < 0: --- False

4 if -10 < 0: --- True

5 y = --10

y = 10

6 if 0 == 0: --- True

7 return 10

3. hex(225) = 'E1'

3 def hex(number=225)

4 if 225 == 0: --- False

6 res = ''

7 while 225 > 0: --- True

8 digit = 225 % 16

digit = 1

9 res = DIGITS[1] + ''

res = '1'

10 number = 225 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + '1'

res = 'E1'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'E1'

4. square\_equal(0, -5, 8) = [1.6]

3 def square\_equal(a=0, b=-5, c=8)

4 if 0 != 0: --- False

14 else:

15 if -5 != 0: --- True

16 return [1.6]

5. square\_equal(-20, 2, -12) = []

3 def square\_equal(a=-20, b=2, c=-12)

4 if -20 != 0: --- True

5 D = 2\*2 - 4\*-20\*-12

D = -956

6 if -956 > 0: --- False

10 elif -956 == 0: --- False

12 else:

13 return []

6. findmax([98, 34, 74, 7, 56, 76]) = 98

1 def findmax(items=[98, 34, 74, 7, 56, 76])

2 if len([98, 34, 74, 7, 56, 76]) == 0: --- False

4 m = items[0]

m = 98

5 i = 1

6 while 1 < len([98, 34, 74, 7, 56, 76]): --- True

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

9 i = 1 + 1

i = 2

6 while 2 < len([98, 34, 74, 7, 56, 76]): --- True

7 if 98 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([98, 34, 74, 7, 56, 76]): --- True

7 if 98 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([98, 34, 74, 7, 56, 76]): --- True

7 if 98 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([98, 34, 74, 7, 56, 76]): --- True

7 if 98 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([98, 34, 74, 7, 56, 76]): --- False

10 return 98

7. unique([-29, -29, -68, -15]) = [-29, -68, -15]

1 def unique(items=[-29, -29, -68, -15])

2 res = []

3 i = 0

4 while 0 < len([-29, -29, -68, -15]): --- True

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

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

res = [-29]

7 i = 0 + 1

i = 1

4 while 1 < len([-29, -29, -68, -15]): --- True

5 if len([-29]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-29, -29, -68, -15]): --- True

5 if len([-29]) == 0 or res[-1] != items[2]: --- True

6 res = [-29] + [items[2]]

res = [-29, -68]

7 i = 2 + 1

i = 3

4 while 3 < len([-29, -29, -68, -15]): --- True

5 if len([-29, -68]) == 0 or res[-1] != items[3]: --- True

6 res = [-29, -68] + [items[3]]

res = [-29, -68, -15]

7 i = 3 + 1

i = 4

4 while 4 < len([-29, -29, -68, -15]): --- False

8 return [-29, -68, -15]

8. join(':', [98, 66, 28, 81]) = '98:66:28:81'

1 def join(sep=:, items=[98, 66, 28, 81])

2 res = ''

3 if len([98, 66, 28, 81]) > 0: --- True

4 res = str(items[0])

res = '98'

5 items = items[1:]

items = [66, 28, 81]

6 while len([66, 28, 81]) > 0: --- True

7 res = '98' + ':' + str(items[0])

res = '98:66'

8 items = items[1:]

items = [28, 81]

6 while len([28, 81]) > 0: --- True

7 res = '98:66' + ':' + str(items[0])

res = '98:66:28'

8 items = items[1:]

items = [81]

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

7 res = '98:66:28' + ':' + str(items[0])

res = '98:66:28:81'

8 items = items[1:]

items = []

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

9 return '98:66:28:81'

Вариант: 2-1-30

1. gcd(-88, -80) = 8

1 def gcd(x=-88, y=-80)

2 if -88 < 0: --- True

3 x = --88

x = 88

4 if -80 < 0: --- True

5 y = --80

y = 80

6 if 88 == 0: --- False

8 while 80 != 0: --- True

9 rem = 88 % 80

rem = 8

10 x = 80

11 y = 8

8 while 8 != 0: --- True

9 rem = 80 % 8

rem = 0

10 x = 8

11 y = 0

8 while 0 != 0: --- False

12 return 8

2. gcd(0, 44) = 44

1 def gcd(x=0, y=44)

2 if 0 < 0: --- False

4 if 44 < 0: --- False

6 if 0 == 0: --- True

7 return 44

3. hex(170) = 'AA'

3 def hex(number=170)

4 if 170 == 0: --- False

6 res = ''

7 while 170 > 0: --- True

8 digit = 170 % 16

digit = 10

9 res = DIGITS[10] + ''

res = 'A'

10 number = 170 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + 'A'

res = 'AA'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'AA'

4. square\_equal(-50, 55, -9) = [0.9, 0.2]

3 def square\_equal(a=-50, b=55, c=-9)

4 if -50 != 0: --- True

5 D = 55\*55 - 4\*-50\*-9

D = 1225

6 if 1225 > 0: --- True

7 x1 = (-55 - sqrt(1225)) / (2\*-50)

x1 = 0.9

8 x2 = (-55 + sqrt(1225)) / (2\*-50)

x2 = 0.2

9 return [0.9, 0.2]

5. square\_equal(-29, -71, -64) = []

3 def square\_equal(a=-29, b=-71, c=-64)

4 if -29 != 0: --- True

5 D = -71\*-71 - 4\*-29\*-64

D = -2383

6 if -2383 > 0: --- False

10 elif -2383 == 0: --- False

12 else:

13 return []

6. findmax([-14, 57, 23, -42, 13]) = 57

1 def findmax(items=[-14, 57, 23, -42, 13])

2 if len([-14, 57, 23, -42, 13]) == 0: --- False

4 m = items[0]

m = -14

5 i = 1

6 while 1 < len([-14, 57, 23, -42, 13]): --- True

7 if -14 < items[1]: --- True

8 m = items[1]

m = 57

9 i = 1 + 1

i = 2

6 while 2 < len([-14, 57, 23, -42, 13]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([-14, 57, 23, -42, 13]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([-14, 57, 23, -42, 13]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([-14, 57, 23, -42, 13]): --- False

10 return 57

7. unique([-11, -11, -14, -70]) = [-11, -14, -70]

1 def unique(items=[-11, -11, -14, -70])

2 res = []

3 i = 0

4 while 0 < len([-11, -11, -14, -70]): --- True

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

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

res = [-11]

7 i = 0 + 1

i = 1

4 while 1 < len([-11, -11, -14, -70]): --- True

5 if len([-11]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-11, -11, -14, -70]): --- True

5 if len([-11]) == 0 or res[-1] != items[2]: --- True

6 res = [-11] + [items[2]]

res = [-11, -14]

7 i = 2 + 1

i = 3

4 while 3 < len([-11, -11, -14, -70]): --- True

5 if len([-11, -14]) == 0 or res[-1] != items[3]: --- True

6 res = [-11, -14] + [items[3]]

res = [-11, -14, -70]

7 i = 3 + 1

i = 4

4 while 4 < len([-11, -11, -14, -70]): --- False

8 return [-11, -14, -70]

8. join(';', [67, 90, 99]) = '67;90;99'

1 def join(sep=;, items=[67, 90, 99])

2 res = ''

3 if len([67, 90, 99]) > 0: --- True

4 res = str(items[0])

res = '67'

5 items = items[1:]

items = [90, 99]

6 while len([90, 99]) > 0: --- True

7 res = '67' + ';' + str(items[0])

res = '67;90'

8 items = items[1:]

items = [99]

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

7 res = '67;90' + ';' + str(items[0])

res = '67;90;99'

8 items = items[1:]

items = []

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

9 return '67;90;99'

Вариант: 2-1-31

1. gcd(-48, 20) = 4

1 def gcd(x=-48, y=20)

2 if -48 < 0: --- True

3 x = --48

x = 48

4 if 20 < 0: --- False

6 if 48 == 0: --- False

8 while 20 != 0: --- True

9 rem = 48 % 20

rem = 8

10 x = 20

11 y = 8

8 while 8 != 0: --- True

9 rem = 20 % 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(-66, 0) = 66

1 def gcd(x=-66, y=0)

2 if -66 < 0: --- True

3 x = --66

x = 66

4 if 0 < 0: --- False

6 if 66 == 0: --- False

8 while 0 != 0: --- False

12 return 66

3. hex(173) = 'AD'

3 def hex(number=173)

4 if 173 == 0: --- False

6 res = ''

7 while 173 > 0: --- True

8 digit = 173 % 16

digit = 13

9 res = DIGITS[13] + ''

res = 'D'

10 number = 173 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + 'D'

res = 'AD'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'AD'

4. square\_equal(0, -25, -45) = [-1.8]

3 def square\_equal(a=0, b=-25, c=-45)

4 if 0 != 0: --- False

14 else:

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

16 return [-1.8]

5. square\_equal(75, 96, 39) = []

3 def square\_equal(a=75, b=96, c=39)

4 if 75 != 0: --- True

5 D = 96\*96 - 4\*75\*39

D = -2484

6 if -2484 > 0: --- False

10 elif -2484 == 0: --- False

12 else:

13 return []

6. findmax([-46, 55, 15, -32]) = 55

1 def findmax(items=[-46, 55, 15, -32])

2 if len([-46, 55, 15, -32]) == 0: --- False

4 m = items[0]

m = -46

5 i = 1

6 while 1 < len([-46, 55, 15, -32]): --- True

7 if -46 < items[1]: --- True

8 m = items[1]

m = 55

9 i = 1 + 1

i = 2

6 while 2 < len([-46, 55, 15, -32]): --- True

7 if 55 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-46, 55, 15, -32]): --- True

7 if 55 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-46, 55, 15, -32]): --- False

10 return 55

7. unique([-74, -74, -87, -74]) = [-74, -87, -74]

1 def unique(items=[-74, -74, -87, -74])

2 res = []

3 i = 0

4 while 0 < len([-74, -74, -87, -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, -87, -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, -87, -74]): --- True

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

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

res = [-74, -87]

7 i = 2 + 1

i = 3

4 while 3 < len([-74, -74, -87, -74]): --- True

5 if len([-74, -87]) == 0 or res[-1] != items[3]: --- True

6 res = [-74, -87] + [items[3]]

res = [-74, -87, -74]

7 i = 3 + 1

i = 4

4 while 4 < len([-74, -74, -87, -74]): --- False

8 return [-74, -87, -74]

8. join(';', [50, 49, 16, 81]) = '50;49;16;81'

1 def join(sep=;, items=[50, 49, 16, 81])

2 res = ''

3 if len([50, 49, 16, 81]) > 0: --- True

4 res = str(items[0])

res = '50'

5 items = items[1:]

items = [49, 16, 81]

6 while len([49, 16, 81]) > 0: --- True

7 res = '50' + ';' + str(items[0])

res = '50;49'

8 items = items[1:]

items = [16, 81]

6 while len([16, 81]) > 0: --- True

7 res = '50;49' + ';' + str(items[0])

res = '50;49;16'

8 items = items[1:]

items = [81]

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

7 res = '50;49;16' + ';' + str(items[0])

res = '50;49;16;81'

8 items = items[1:]

items = []

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

9 return '50;49;16;81'

Вариант: 2-1-32

1. gcd(39, 30) = 3

1 def gcd(x=39, y=30)

2 if 39 < 0: --- False

4 if 30 < 0: --- False

6 if 39 == 0: --- False

8 while 30 != 0: --- True

9 rem = 39 % 30

rem = 9

10 x = 30

11 y = 9

8 while 9 != 0: --- True

9 rem = 30 % 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(-86, 0) = 86

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

2 if -86 < 0: --- True

3 x = --86

x = 86

4 if 0 < 0: --- False

6 if 86 == 0: --- False

8 while 0 != 0: --- False

12 return 86

3. hex(167) = 'A7'

3 def hex(number=167)

4 if 167 == 0: --- False

6 res = ''

7 while 167 > 0: --- True

8 digit = 167 % 16

digit = 7

9 res = DIGITS[7] + ''

res = '7'

10 number = 167 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + '7'

res = 'A7'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'A7'

4. square\_equal(-15, 48, 63) = [4.2, -1.0]

3 def square\_equal(a=-15, b=48, c=63)

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

5 D = 48\*48 - 4\*-15\*63

D = 6084

6 if 6084 > 0: --- True

7 x1 = (-48 - sqrt(6084)) / (2\*-15)

x1 = 4.2

8 x2 = (-48 + sqrt(6084)) / (2\*-15)

x2 = -1.0

9 return [4.2, -1.0]

5. square\_equal(50, 55, 23) = []

3 def square\_equal(a=50, b=55, c=23)

4 if 50 != 0: --- True

5 D = 55\*55 - 4\*50\*23

D = -1575

6 if -1575 > 0: --- False

10 elif -1575 == 0: --- False

12 else:

13 return []

6. findmax([42, 42, 75, 93]) = 93

1 def findmax(items=[42, 42, 75, 93])

2 if len([42, 42, 75, 93]) == 0: --- False

4 m = items[0]

m = 42

5 i = 1

6 while 1 < len([42, 42, 75, 93]): --- True

7 if 42 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([42, 42, 75, 93]): --- True

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

8 m = items[2]

m = 75

9 i = 2 + 1

i = 3

6 while 3 < len([42, 42, 75, 93]): --- True

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

8 m = items[3]

m = 93

9 i = 3 + 1

i = 4

6 while 4 < len([42, 42, 75, 93]): --- False

10 return 93

7. unique([-41, -77, -77, -64]) = [-41, -77, -64]

1 def unique(items=[-41, -77, -77, -64])

2 res = []

3 i = 0

4 while 0 < len([-41, -77, -77, -64]): --- True

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

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

res = [-41]

7 i = 0 + 1

i = 1

4 while 1 < len([-41, -77, -77, -64]): --- True

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

6 res = [-41] + [items[1]]

res = [-41, -77]

7 i = 1 + 1

i = 2

4 while 2 < len([-41, -77, -77, -64]): --- True

5 if len([-41, -77]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-41, -77, -77, -64]): --- True

5 if len([-41, -77]) == 0 or res[-1] != items[3]: --- True

6 res = [-41, -77] + [items[3]]

res = [-41, -77, -64]

7 i = 3 + 1

i = 4

4 while 4 < len([-41, -77, -77, -64]): --- False

8 return [-41, -77, -64]

8. join('+', [76, 42, 52, 33]) = '76+42+52+33'

1 def join(sep=+, items=[76, 42, 52, 33])

2 res = ''

3 if len([76, 42, 52, 33]) > 0: --- True

4 res = str(items[0])

res = '76'

5 items = items[1:]

items = [42, 52, 33]

6 while len([42, 52, 33]) > 0: --- True

7 res = '76' + '+' + str(items[0])

res = '76+42'

8 items = items[1:]

items = [52, 33]

6 while len([52, 33]) > 0: --- True

7 res = '76+42' + '+' + str(items[0])

res = '76+42+52'

8 items = items[1:]

items = [33]

6 while len([33]) > 0: --- True

7 res = '76+42+52' + '+' + str(items[0])

res = '76+42+52+33'

8 items = items[1:]

items = []

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

9 return '76+42+52+33'

Вариант: 2-1-33

1. gcd(-75, 45) = 15

1 def gcd(x=-75, y=45)

2 if -75 < 0: --- True

3 x = --75

x = 75

4 if 45 < 0: --- False

6 if 75 == 0: --- False

8 while 45 != 0: --- True

9 rem = 75 % 45

rem = 30

10 x = 45

11 y = 30

8 while 30 != 0: --- True

9 rem = 45 % 30

rem = 15

10 x = 30

11 y = 15

8 while 15 != 0: --- True

9 rem = 30 % 15

rem = 0

10 x = 15

11 y = 0

8 while 0 != 0: --- False

12 return 15

2. gcd(0, -16) = 16

1 def gcd(x=0, y=-16)

2 if 0 < 0: --- False

4 if -16 < 0: --- True

5 y = --16

y = 16

6 if 0 == 0: --- True

7 return 16

3. hex(199) = 'C7'

3 def hex(number=199)

4 if 199 == 0: --- False

6 res = ''

7 while 199 > 0: --- True

8 digit = 199 % 16

digit = 7

9 res = DIGITS[7] + ''

res = '7'

10 number = 199 // 16

number = 12

7 while 12 > 0: --- True

8 digit = 12 % 16

digit = 12

9 res = DIGITS[12] + '7'

res = 'C7'

10 number = 12 // 16

number = 0

7 while 0 > 0: --- False

11 return 'C7'

4. square\_equal(40, -56, -6) = [-0.1, 1.5]

3 def square\_equal(a=40, b=-56, c=-6)

4 if 40 != 0: --- True

5 D = -56\*-56 - 4\*40\*-6

D = 4096

6 if 4096 > 0: --- True

7 x1 = (--56 - sqrt(4096)) / (2\*40)

x1 = -0.1

8 x2 = (--56 + sqrt(4096)) / (2\*40)

x2 = 1.5

9 return [-0.1, 1.5]

5. square\_equal(52, 87, 82) = []

3 def square\_equal(a=52, b=87, c=82)

4 if 52 != 0: --- True

5 D = 87\*87 - 4\*52\*82

D = -9487

6 if -9487 > 0: --- False

10 elif -9487 == 0: --- False

12 else:

13 return []

6. findmax([43, 8, 47, -79, 91]) = 91

1 def findmax(items=[43, 8, 47, -79, 91])

2 if len([43, 8, 47, -79, 91]) == 0: --- False

4 m = items[0]

m = 43

5 i = 1

6 while 1 < len([43, 8, 47, -79, 91]): --- True

7 if 43 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([43, 8, 47, -79, 91]): --- True

7 if 43 < items[2]: --- True

8 m = items[2]

m = 47

9 i = 2 + 1

i = 3

6 while 3 < len([43, 8, 47, -79, 91]): --- True

7 if 47 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([43, 8, 47, -79, 91]): --- True

7 if 47 < items[4]: --- True

8 m = items[4]

m = 91

9 i = 4 + 1

i = 5

6 while 5 < len([43, 8, 47, -79, 91]): --- False

10 return 91

7. unique([83, 83, -45]) = [83, -45]

1 def unique(items=[83, 83, -45])

2 res = []

3 i = 0

4 while 0 < len([83, 83, -45]): --- True

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

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

res = [83]

7 i = 0 + 1

i = 1

4 while 1 < len([83, 83, -45]): --- True

5 if len([83]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([83, 83, -45]): --- True

5 if len([83]) == 0 or res[-1] != items[2]: --- True

6 res = [83] + [items[2]]

res = [83, -45]

7 i = 2 + 1

i = 3

4 while 3 < len([83, 83, -45]): --- False

8 return [83, -45]

8. join(';', [7, 38, 77]) = '7;38;77'

1 def join(sep=;, items=[7, 38, 77])

2 res = ''

3 if len([7, 38, 77]) > 0: --- True

4 res = str(items[0])

res = '7'

5 items = items[1:]

items = [38, 77]

6 while len([38, 77]) > 0: --- True

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

res = '7;38'

8 items = items[1:]

items = [77]

6 while len([77]) > 0: --- True

7 res = '7;38' + ';' + str(items[0])

res = '7;38;77'

8 items = items[1:]

items = []

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

9 return '7;38;77'

Вариант: 2-1-34

1. gcd(75, -27) = 3

1 def gcd(x=75, y=-27)

2 if 75 < 0: --- False

4 if -27 < 0: --- True

5 y = --27

y = 27

6 if 75 == 0: --- False

8 while 27 != 0: --- True

9 rem = 75 % 27

rem = 21

10 x = 27

11 y = 21

8 while 21 != 0: --- True

9 rem = 27 % 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(0, -72) = 72

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

2 if 0 < 0: --- False

4 if -72 < 0: --- True

5 y = --72

y = 72

6 if 0 == 0: --- True

7 return 72

3. hex(201) = 'C9'

3 def hex(number=201)

4 if 201 == 0: --- False

6 res = ''

7 while 201 > 0: --- True

8 digit = 201 % 16

digit = 9

9 res = DIGITS[9] + ''

res = '9'

10 number = 201 // 16

number = 12

7 while 12 > 0: --- True

8 digit = 12 % 16

digit = 12

9 res = DIGITS[12] + '9'

res = 'C9'

10 number = 12 // 16

number = 0

7 while 0 > 0: --- False

11 return 'C9'

4. square\_equal(-2, 60, -58) = [29.0, 1.0]

3 def square\_equal(a=-2, b=60, c=-58)

4 if -2 != 0: --- True

5 D = 60\*60 - 4\*-2\*-58

D = 3136

6 if 3136 > 0: --- True

7 x1 = (-60 - sqrt(3136)) / (2\*-2)

x1 = 29.0

8 x2 = (-60 + sqrt(3136)) / (2\*-2)

x2 = 1.0

9 return [29.0, 1.0]

5. square\_equal(-14, -57, -63) = []

3 def square\_equal(a=-14, b=-57, c=-63)

4 if -14 != 0: --- True

5 D = -57\*-57 - 4\*-14\*-63

D = -279

6 if -279 > 0: --- False

10 elif -279 == 0: --- False

12 else:

13 return []

6. findmax([-9, -47, -11, 22]) = 22

1 def findmax(items=[-9, -47, -11, 22])

2 if len([-9, -47, -11, 22]) == 0: --- False

4 m = items[0]

m = -9

5 i = 1

6 while 1 < len([-9, -47, -11, 22]): --- True

7 if -9 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([-9, -47, -11, 22]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([-9, -47, -11, 22]): --- True

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

8 m = items[3]

m = 22

9 i = 3 + 1

i = 4

6 while 4 < len([-9, -47, -11, 22]): --- False

10 return 22

7. unique([-5, -5, -53]) = [-5, -53]

1 def unique(items=[-5, -5, -53])

2 res = []

3 i = 0

4 while 0 < len([-5, -5, -53]): --- 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, -53]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([-5, -5, -53]): --- True

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

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

res = [-5, -53]

7 i = 2 + 1

i = 3

4 while 3 < len([-5, -5, -53]): --- False

8 return [-5, -53]

8. join(':', [63, 24, 19, 37]) = '63:24:19:37'

1 def join(sep=:, items=[63, 24, 19, 37])

2 res = ''

3 if len([63, 24, 19, 37]) > 0: --- True

4 res = str(items[0])

res = '63'

5 items = items[1:]

items = [24, 19, 37]

6 while len([24, 19, 37]) > 0: --- True

7 res = '63' + ':' + str(items[0])

res = '63:24'

8 items = items[1:]

items = [19, 37]

6 while len([19, 37]) > 0: --- True

7 res = '63:24' + ':' + str(items[0])

res = '63:24:19'

8 items = items[1:]

items = [37]

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

7 res = '63:24:19' + ':' + str(items[0])

res = '63:24:19:37'

8 items = items[1:]

items = []

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

9 return '63:24:19:37'

Вариант: 2-1-35

1. gcd(45, -50) = 5

1 def gcd(x=45, y=-50)

2 if 45 < 0: --- False

4 if -50 < 0: --- True

5 y = --50

y = 50

6 if 45 == 0: --- False

8 while 50 != 0: --- True

9 rem = 45 % 50

rem = 45

10 x = 50

11 y = 45

8 while 45 != 0: --- True

9 rem = 50 % 45

rem = 5

10 x = 45

11 y = 5

8 while 5 != 0: --- True

9 rem = 45 % 5

rem = 0

10 x = 5

11 y = 0

8 while 0 != 0: --- False

12 return 5

2. gcd(-89, 0) = 89

1 def gcd(x=-89, y=0)

2 if -89 < 0: --- True

3 x = --89

x = 89

4 if 0 < 0: --- False

6 if 89 == 0: --- False

8 while 0 != 0: --- False

12 return 89

3. hex(174) = 'AE'

3 def hex(number=174)

4 if 174 == 0: --- False

6 res = ''

7 while 174 > 0: --- True

8 digit = 174 % 16

digit = 14

9 res = DIGITS[14] + ''

res = 'E'

10 number = 174 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + 'E'

res = 'AE'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'AE'

4. square\_equal(4, 30, 56) = [-4.0, -3.5]

3 def square\_equal(a=4, b=30, c=56)

4 if 4 != 0: --- True

5 D = 30\*30 - 4\*4\*56

D = 4

6 if 4 > 0: --- True

7 x1 = (-30 - sqrt(4)) / (2\*4)

x1 = -4.0

8 x2 = (-30 + sqrt(4)) / (2\*4)

x2 = -3.5

9 return [-4.0, -3.5]

5. square\_equal(-99, -14, -49) = []

3 def square\_equal(a=-99, b=-14, c=-49)

4 if -99 != 0: --- True

5 D = -14\*-14 - 4\*-99\*-49

D = -19208

6 if -19208 > 0: --- False

10 elif -19208 == 0: --- False

12 else:

13 return []

6. findmax([74, 93, -69, 37, 28]) = 93

1 def findmax(items=[74, 93, -69, 37, 28])

2 if len([74, 93, -69, 37, 28]) == 0: --- False

4 m = items[0]

m = 74

5 i = 1

6 while 1 < len([74, 93, -69, 37, 28]): --- True

7 if 74 < items[1]: --- True

8 m = items[1]

m = 93

9 i = 1 + 1

i = 2

6 while 2 < len([74, 93, -69, 37, 28]): --- True

7 if 93 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([74, 93, -69, 37, 28]): --- True

7 if 93 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([74, 93, -69, 37, 28]): --- True

7 if 93 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([74, 93, -69, 37, 28]): --- False

10 return 93

7. unique([27, 27, -74, -79]) = [27, -74, -79]

1 def unique(items=[27, 27, -74, -79])

2 res = []

3 i = 0

4 while 0 < len([27, 27, -74, -79]): --- True

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

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

res = [27]

7 i = 0 + 1

i = 1

4 while 1 < len([27, 27, -74, -79]): --- True

5 if len([27]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([27, 27, -74, -79]): --- True

5 if len([27]) == 0 or res[-1] != items[2]: --- True

6 res = [27] + [items[2]]

res = [27, -74]

7 i = 2 + 1

i = 3

4 while 3 < len([27, 27, -74, -79]): --- True

5 if len([27, -74]) == 0 or res[-1] != items[3]: --- True

6 res = [27, -74] + [items[3]]

res = [27, -74, -79]

7 i = 3 + 1

i = 4

4 while 4 < len([27, 27, -74, -79]): --- False

8 return [27, -74, -79]

8. join(',', [91, 97, 98]) = '91,97,98'

1 def join(sep=,, items=[91, 97, 98])

2 res = ''

3 if len([91, 97, 98]) > 0: --- True

4 res = str(items[0])

res = '91'

5 items = items[1:]

items = [97, 98]

6 while len([97, 98]) > 0: --- True

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

res = '91,97'

8 items = items[1:]

items = [98]

6 while len([98]) > 0: --- True

7 res = '91,97' + ',' + str(items[0])

res = '91,97,98'

8 items = items[1:]

items = []

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

9 return '91,97,98'

Вариант: 2-1-36

1. gcd(-54, -51) = 3

1 def gcd(x=-54, y=-51)

2 if -54 < 0: --- True

3 x = --54

x = 54

4 if -51 < 0: --- True

5 y = --51

y = 51

6 if 54 == 0: --- False

8 while 51 != 0: --- True

9 rem = 54 % 51

rem = 3

10 x = 51

11 y = 3

8 while 3 != 0: --- True

9 rem = 51 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(0, 76) = 76

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

2 if 0 < 0: --- False

4 if 76 < 0: --- False

6 if 0 == 0: --- True

7 return 76

3. hex(162) = 'A2'

3 def hex(number=162)

4 if 162 == 0: --- False

6 res = ''

7 while 162 > 0: --- True

8 digit = 162 % 16

digit = 2

9 res = DIGITS[2] + ''

res = '2'

10 number = 162 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + '2'

res = 'A2'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'A2'

4. square\_equal(-20, -37, 0) = [-0.0, -1.85]

3 def square\_equal(a=-20, b=-37, c=0)

4 if -20 != 0: --- True

5 D = -37\*-37 - 4\*-20\*0

D = 1369

6 if 1369 > 0: --- True

7 x1 = (--37 - sqrt(1369)) / (2\*-20)

x1 = -0.0

8 x2 = (--37 + sqrt(1369)) / (2\*-20)

x2 = -1.85

9 return [-0.0, -1.85]

5. square\_equal(40, -49, 80) = []

3 def square\_equal(a=40, b=-49, c=80)

4 if 40 != 0: --- True

5 D = -49\*-49 - 4\*40\*80

D = -10399

6 if -10399 > 0: --- False

10 elif -10399 == 0: --- False

12 else:

13 return []

6. findmax([87, 57, -96, 62, 53]) = 87

1 def findmax(items=[87, 57, -96, 62, 53])

2 if len([87, 57, -96, 62, 53]) == 0: --- False

4 m = items[0]

m = 87

5 i = 1

6 while 1 < len([87, 57, -96, 62, 53]): --- True

7 if 87 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([87, 57, -96, 62, 53]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([87, 57, -96, 62, 53]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([87, 57, -96, 62, 53]): --- True

7 if 87 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([87, 57, -96, 62, 53]): --- False

10 return 87

7. unique([-25, -25, -36]) = [-25, -36]

1 def unique(items=[-25, -25, -36])

2 res = []

3 i = 0

4 while 0 < len([-25, -25, -36]): --- 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, -25, -36]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([-25, -25, -36]): --- True

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

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

res = [-25, -36]

7 i = 2 + 1

i = 3

4 while 3 < len([-25, -25, -36]): --- False

8 return [-25, -36]

8. join('+', [99, 12, 65]) = '99+12+65'

1 def join(sep=+, items=[99, 12, 65])

2 res = ''

3 if len([99, 12, 65]) > 0: --- True

4 res = str(items[0])

res = '99'

5 items = items[1:]

items = [12, 65]

6 while len([12, 65]) > 0: --- True

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

res = '99+12'

8 items = items[1:]

items = [65]

6 while len([65]) > 0: --- True

7 res = '99+12' + '+' + str(items[0])

res = '99+12+65'

8 items = items[1:]

items = []

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

9 return '99+12+65'

Вариант: 2-1-37

1. gcd(-60, -51) = 3

1 def gcd(x=-60, y=-51)

2 if -60 < 0: --- True

3 x = --60

x = 60

4 if -51 < 0: --- True

5 y = --51

y = 51

6 if 60 == 0: --- False

8 while 51 != 0: --- True

9 rem = 60 % 51

rem = 9

10 x = 51

11 y = 9

8 while 9 != 0: --- True

9 rem = 51 % 9

rem = 6

10 x = 9

11 y = 6

8 while 6 != 0: --- True

9 rem = 9 % 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(84, 0) = 84

1 def gcd(x=84, y=0)

2 if 84 < 0: --- False

4 if 0 < 0: --- False

6 if 84 == 0: --- False

8 while 0 != 0: --- False

12 return 84

3. hex(172) = 'AC'

3 def hex(number=172)

4 if 172 == 0: --- False

6 res = ''

7 while 172 > 0: --- True

8 digit = 172 % 16

digit = 12

9 res = DIGITS[12] + ''

res = 'C'

10 number = 172 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + 'C'

res = 'AC'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'AC'

4. square\_equal(5, -19, 14) = [1.0, 2.8]

3 def square\_equal(a=5, b=-19, c=14)

4 if 5 != 0: --- True

5 D = -19\*-19 - 4\*5\*14

D = 81

6 if 81 > 0: --- True

7 x1 = (--19 - sqrt(81)) / (2\*5)

x1 = 1.0

8 x2 = (--19 + sqrt(81)) / (2\*5)

x2 = 2.8

9 return [1.0, 2.8]

5. square\_equal(61, 7, 11) = []

3 def square\_equal(a=61, b=7, c=11)

4 if 61 != 0: --- True

5 D = 7\*7 - 4\*61\*11

D = -2635

6 if -2635 > 0: --- False

10 elif -2635 == 0: --- False

12 else:

13 return []

6. findmax([9, 0, -93, 24]) = 24

1 def findmax(items=[9, 0, -93, 24])

2 if len([9, 0, -93, 24]) == 0: --- False

4 m = items[0]

m = 9

5 i = 1

6 while 1 < len([9, 0, -93, 24]): --- True

7 if 9 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([9, 0, -93, 24]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([9, 0, -93, 24]): --- True

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

8 m = items[3]

m = 24

9 i = 3 + 1

i = 4

6 while 4 < len([9, 0, -93, 24]): --- False

10 return 24

7. unique([61, 59, 61, 61]) = [61, 59, 61]

1 def unique(items=[61, 59, 61, 61])

2 res = []

3 i = 0

4 while 0 < len([61, 59, 61, 61]): --- 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, 59, 61, 61]): --- True

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

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

res = [61, 59]

7 i = 1 + 1

i = 2

4 while 2 < len([61, 59, 61, 61]): --- True

5 if len([61, 59]) == 0 or res[-1] != items[2]: --- True

6 res = [61, 59] + [items[2]]

res = [61, 59, 61]

7 i = 2 + 1

i = 3

4 while 3 < len([61, 59, 61, 61]): --- True

5 if len([61, 59, 61]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([61, 59, 61, 61]): --- False

8 return [61, 59, 61]

8. join('+', [75, 55, 94, 38]) = '75+55+94+38'

1 def join(sep=+, items=[75, 55, 94, 38])

2 res = ''

3 if len([75, 55, 94, 38]) > 0: --- True

4 res = str(items[0])

res = '75'

5 items = items[1:]

items = [55, 94, 38]

6 while len([55, 94, 38]) > 0: --- True

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

res = '75+55'

8 items = items[1:]

items = [94, 38]

6 while len([94, 38]) > 0: --- True

7 res = '75+55' + '+' + str(items[0])

res = '75+55+94'

8 items = items[1:]

items = [38]

6 while len([38]) > 0: --- True

7 res = '75+55+94' + '+' + str(items[0])

res = '75+55+94+38'

8 items = items[1:]

items = []

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

9 return '75+55+94+38'

Вариант: 2-1-38

1. gcd(65, 39) = 13

1 def gcd(x=65, y=39)

2 if 65 < 0: --- False

4 if 39 < 0: --- False

6 if 65 == 0: --- False

8 while 39 != 0: --- True

9 rem = 65 % 39

rem = 26

10 x = 39

11 y = 26

8 while 26 != 0: --- True

9 rem = 39 % 26

rem = 13

10 x = 26

11 y = 13

8 while 13 != 0: --- True

9 rem = 26 % 13

rem = 0

10 x = 13

11 y = 0

8 while 0 != 0: --- False

12 return 13

2. gcd(0, -85) = 85

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

2 if 0 < 0: --- False

4 if -85 < 0: --- True

5 y = --85

y = 85

6 if 0 == 0: --- True

7 return 85

3. hex(181) = 'B5'

3 def hex(number=181)

4 if 181 == 0: --- False

6 res = ''

7 while 181 > 0: --- True

8 digit = 181 % 16

digit = 5

9 res = DIGITS[5] + ''

res = '5'

10 number = 181 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + '5'

res = 'B5'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'B5'

4. square\_equal(0, 10, -94) = [9.4]

3 def square\_equal(a=0, b=10, c=-94)

4 if 0 != 0: --- False

14 else:

15 if 10 != 0: --- True

16 return [9.4]

5. square\_equal(89, 84, 95) = []

3 def square\_equal(a=89, b=84, c=95)

4 if 89 != 0: --- True

5 D = 84\*84 - 4\*89\*95

D = -26764

6 if -26764 > 0: --- False

10 elif -26764 == 0: --- False

12 else:

13 return []

6. findmax([-93, -27, -95, 69, -16]) = 69

1 def findmax(items=[-93, -27, -95, 69, -16])

2 if len([-93, -27, -95, 69, -16]) == 0: --- False

4 m = items[0]

m = -93

5 i = 1

6 while 1 < len([-93, -27, -95, 69, -16]): --- True

7 if -93 < items[1]: --- True

8 m = items[1]

m = -27

9 i = 1 + 1

i = 2

6 while 2 < len([-93, -27, -95, 69, -16]): --- True

7 if -27 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-93, -27, -95, 69, -16]): --- True

7 if -27 < items[3]: --- True

8 m = items[3]

m = 69

9 i = 3 + 1

i = 4

6 while 4 < len([-93, -27, -95, 69, -16]): --- True

7 if 69 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-93, -27, -95, 69, -16]): --- False

10 return 69

7. unique([37, 37, 16, 37]) = [37, 16, 37]

1 def unique(items=[37, 37, 16, 37])

2 res = []

3 i = 0

4 while 0 < len([37, 37, 16, 37]): --- True

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

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

res = [37]

7 i = 0 + 1

i = 1

4 while 1 < len([37, 37, 16, 37]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([37, 37, 16, 37]): --- True

5 if len([37]) == 0 or res[-1] != items[2]: --- True

6 res = [37] + [items[2]]

res = [37, 16]

7 i = 2 + 1

i = 3

4 while 3 < len([37, 37, 16, 37]): --- True

5 if len([37, 16]) == 0 or res[-1] != items[3]: --- True

6 res = [37, 16] + [items[3]]

res = [37, 16, 37]

7 i = 3 + 1

i = 4

4 while 4 < len([37, 37, 16, 37]): --- False

8 return [37, 16, 37]

8. join(',', [7, 52, 89]) = '7,52,89'

1 def join(sep=,, items=[7, 52, 89])

2 res = ''

3 if len([7, 52, 89]) > 0: --- True

4 res = str(items[0])

res = '7'

5 items = items[1:]

items = [52, 89]

6 while len([52, 89]) > 0: --- True

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

res = '7,52'

8 items = items[1:]

items = [89]

6 while len([89]) > 0: --- True

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

res = '7,52,89'

8 items = items[1:]

items = []

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

9 return '7,52,89'

Вариант: 2-1-39

1. gcd(-33, -72) = 3

1 def gcd(x=-33, y=-72)

2 if -33 < 0: --- True

3 x = --33

x = 33

4 if -72 < 0: --- True

5 y = --72

y = 72

6 if 33 == 0: --- False

8 while 72 != 0: --- True

9 rem = 33 % 72

rem = 33

10 x = 72

11 y = 33

8 while 33 != 0: --- True

9 rem = 72 % 33

rem = 6

10 x = 33

11 y = 6

8 while 6 != 0: --- True

9 rem = 33 % 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(0, -38) = 38

1 def gcd(x=0, y=-38)

2 if 0 < 0: --- False

4 if -38 < 0: --- True

5 y = --38

y = 38

6 if 0 == 0: --- True

7 return 38

3. hex(222) = 'DE'

3 def hex(number=222)

4 if 222 == 0: --- False

6 res = ''

7 while 222 > 0: --- True

8 digit = 222 % 16

digit = 14

9 res = DIGITS[14] + ''

res = 'E'

10 number = 222 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + 'E'

res = 'DE'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'DE'

4. square\_equal(3, 81, -84) = [-28.0, 1.0]

3 def square\_equal(a=3, b=81, c=-84)

4 if 3 != 0: --- True

5 D = 81\*81 - 4\*3\*-84

D = 7569

6 if 7569 > 0: --- True

7 x1 = (-81 - sqrt(7569)) / (2\*3)

x1 = -28.0

8 x2 = (-81 + sqrt(7569)) / (2\*3)

x2 = 1.0

9 return [-28.0, 1.0]

5. square\_equal(-87, 30, -24) = []

3 def square\_equal(a=-87, b=30, c=-24)

4 if -87 != 0: --- True

5 D = 30\*30 - 4\*-87\*-24

D = -7452

6 if -7452 > 0: --- False

10 elif -7452 == 0: --- False

12 else:

13 return []

6. findmax([-83, 49, -43, -42]) = 49

1 def findmax(items=[-83, 49, -43, -42])

2 if len([-83, 49, -43, -42]) == 0: --- False

4 m = items[0]

m = -83

5 i = 1

6 while 1 < len([-83, 49, -43, -42]): --- True

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

8 m = items[1]

m = 49

9 i = 1 + 1

i = 2

6 while 2 < len([-83, 49, -43, -42]): --- True

7 if 49 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-83, 49, -43, -42]): --- True

7 if 49 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-83, 49, -43, -42]): --- False

10 return 49

7. unique([46, 46, 98]) = [46, 98]

1 def unique(items=[46, 46, 98])

2 res = []

3 i = 0

4 while 0 < len([46, 46, 98]): --- 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, 98]): --- True

5 if len([46]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([46, 46, 98]): --- True

5 if len([46]) == 0 or res[-1] != items[2]: --- True

6 res = [46] + [items[2]]

res = [46, 98]

7 i = 2 + 1

i = 3

4 while 3 < len([46, 46, 98]): --- False

8 return [46, 98]

8. join(';', [63, 79, 11, 3]) = '63;79;11;3'

1 def join(sep=;, items=[63, 79, 11, 3])

2 res = ''

3 if len([63, 79, 11, 3]) > 0: --- True

4 res = str(items[0])

res = '63'

5 items = items[1:]

items = [79, 11, 3]

6 while len([79, 11, 3]) > 0: --- True

7 res = '63' + ';' + str(items[0])

res = '63;79'

8 items = items[1:]

items = [11, 3]

6 while len([11, 3]) > 0: --- True

7 res = '63;79' + ';' + str(items[0])

res = '63;79;11'

8 items = items[1:]

items = [3]

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

7 res = '63;79;11' + ';' + str(items[0])

res = '63;79;11;3'

8 items = items[1:]

items = []

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

9 return '63;79;11;3'

Вариант: 2-1-40

1. gcd(69, 21) = 3

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

2 if 69 < 0: --- False

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(64, 0) = 64

1 def gcd(x=64, y=0)

2 if 64 < 0: --- False

4 if 0 < 0: --- False

6 if 64 == 0: --- False

8 while 0 != 0: --- False

12 return 64

3. hex(250) = 'FA'

3 def hex(number=250)

4 if 250 == 0: --- False

6 res = ''

7 while 250 > 0: --- True

8 digit = 250 % 16

digit = 10

9 res = DIGITS[10] + ''

res = 'A'

10 number = 250 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + 'A'

res = 'FA'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'FA'

4. square\_equal(1, -12, 27) = [3.0, 9.0]

3 def square\_equal(a=1, b=-12, c=27)

4 if 1 != 0: --- True

5 D = -12\*-12 - 4\*1\*27

D = 36

6 if 36 > 0: --- True

7 x1 = (--12 - sqrt(36)) / (2\*1)

x1 = 3.0

8 x2 = (--12 + sqrt(36)) / (2\*1)

x2 = 9.0

9 return [3.0, 9.0]

5. square\_equal(-90, 29, -59) = []

3 def square\_equal(a=-90, b=29, c=-59)

4 if -90 != 0: --- True

5 D = 29\*29 - 4\*-90\*-59

D = -20399

6 if -20399 > 0: --- False

10 elif -20399 == 0: --- False

12 else:

13 return []

6. findmax([14, 50, -23, 9, 36]) = 50

1 def findmax(items=[14, 50, -23, 9, 36])

2 if len([14, 50, -23, 9, 36]) == 0: --- False

4 m = items[0]

m = 14

5 i = 1

6 while 1 < len([14, 50, -23, 9, 36]): --- True

7 if 14 < items[1]: --- True

8 m = items[1]

m = 50

9 i = 1 + 1

i = 2

6 while 2 < len([14, 50, -23, 9, 36]): --- True

7 if 50 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([14, 50, -23, 9, 36]): --- True

7 if 50 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([14, 50, -23, 9, 36]): --- True

7 if 50 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([14, 50, -23, 9, 36]): --- False

10 return 50

7. unique([38, 38, -32, 57]) = [38, -32, 57]

1 def unique(items=[38, 38, -32, 57])

2 res = []

3 i = 0

4 while 0 < len([38, 38, -32, 57]): --- True

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

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

res = [38]

7 i = 0 + 1

i = 1

4 while 1 < len([38, 38, -32, 57]): --- True

5 if len([38]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([38, 38, -32, 57]): --- True

5 if len([38]) == 0 or res[-1] != items[2]: --- True

6 res = [38] + [items[2]]

res = [38, -32]

7 i = 2 + 1

i = 3

4 while 3 < len([38, 38, -32, 57]): --- True

5 if len([38, -32]) == 0 or res[-1] != items[3]: --- True

6 res = [38, -32] + [items[3]]

res = [38, -32, 57]

7 i = 3 + 1

i = 4

4 while 4 < len([38, 38, -32, 57]): --- False

8 return [38, -32, 57]

8. join(';', [6, 25, 89, 58]) = '6;25;89;58'

1 def join(sep=;, items=[6, 25, 89, 58])

2 res = ''

3 if len([6, 25, 89, 58]) > 0: --- True

4 res = str(items[0])

res = '6'

5 items = items[1:]

items = [25, 89, 58]

6 while len([25, 89, 58]) > 0: --- True

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

res = '6;25'

8 items = items[1:]

items = [89, 58]

6 while len([89, 58]) > 0: --- True

7 res = '6;25' + ';' + str(items[0])

res = '6;25;89'

8 items = items[1:]

items = [58]

6 while len([58]) > 0: --- True

7 res = '6;25;89' + ';' + str(items[0])

res = '6;25;89;58'

8 items = items[1:]

items = []

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

9 return '6;25;89;58'

Вариант: 2-1-41

1. gcd(12, -100) = 4

1 def gcd(x=12, y=-100)

2 if 12 < 0: --- False

4 if -100 < 0: --- True

5 y = --100

y = 100

6 if 12 == 0: --- False

8 while 100 != 0: --- True

9 rem = 12 % 100

rem = 12

10 x = 100

11 y = 12

8 while 12 != 0: --- True

9 rem = 100 % 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(26, 0) = 26

1 def gcd(x=26, y=0)

2 if 26 < 0: --- False

4 if 0 < 0: --- False

6 if 26 == 0: --- False

8 while 0 != 0: --- False

12 return 26

3. hex(232) = 'E8'

3 def hex(number=232)

4 if 232 == 0: --- False

6 res = ''

7 while 232 > 0: --- True

8 digit = 232 % 16

digit = 8

9 res = DIGITS[8] + ''

res = '8'

10 number = 232 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + '8'

res = 'E8'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'E8'

4. square\_equal(-40, 46, -12) = [0.75, 0.4]

3 def square\_equal(a=-40, b=46, c=-12)

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

5 D = 46\*46 - 4\*-40\*-12

D = 196

6 if 196 > 0: --- True

7 x1 = (-46 - sqrt(196)) / (2\*-40)

x1 = 0.75

8 x2 = (-46 + sqrt(196)) / (2\*-40)

x2 = 0.4

9 return [0.75, 0.4]

5. square\_equal(-74, 67, -75) = []

3 def square\_equal(a=-74, b=67, c=-75)

4 if -74 != 0: --- True

5 D = 67\*67 - 4\*-74\*-75

D = -17711

6 if -17711 > 0: --- False

10 elif -17711 == 0: --- False

12 else:

13 return []

6. findmax([-45, -88, -51, 21, -59]) = 21

1 def findmax(items=[-45, -88, -51, 21, -59])

2 if len([-45, -88, -51, 21, -59]) == 0: --- False

4 m = items[0]

m = -45

5 i = 1

6 while 1 < len([-45, -88, -51, 21, -59]): --- True

7 if -45 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([-45, -88, -51, 21, -59]): --- True

7 if -45 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-45, -88, -51, 21, -59]): --- True

7 if -45 < items[3]: --- True

8 m = items[3]

m = 21

9 i = 3 + 1

i = 4

6 while 4 < len([-45, -88, -51, 21, -59]): --- True

7 if 21 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-45, -88, -51, 21, -59]): --- False

10 return 21

7. unique([-61, -61, -41, -33]) = [-61, -41, -33]

1 def unique(items=[-61, -61, -41, -33])

2 res = []

3 i = 0

4 while 0 < len([-61, -61, -41, -33]): --- 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, -61, -41, -33]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([-61, -61, -41, -33]): --- True

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

6 res = [-61] + [items[2]]

res = [-61, -41]

7 i = 2 + 1

i = 3

4 while 3 < len([-61, -61, -41, -33]): --- True

5 if len([-61, -41]) == 0 or res[-1] != items[3]: --- True

6 res = [-61, -41] + [items[3]]

res = [-61, -41, -33]

7 i = 3 + 1

i = 4

4 while 4 < len([-61, -61, -41, -33]): --- False

8 return [-61, -41, -33]

8. join(',', [35, 92, 27, 93]) = '35,92,27,93'

1 def join(sep=,, items=[35, 92, 27, 93])

2 res = ''

3 if len([35, 92, 27, 93]) > 0: --- True

4 res = str(items[0])

res = '35'

5 items = items[1:]

items = [92, 27, 93]

6 while len([92, 27, 93]) > 0: --- True

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

res = '35,92'

8 items = items[1:]

items = [27, 93]

6 while len([27, 93]) > 0: --- True

7 res = '35,92' + ',' + str(items[0])

res = '35,92,27'

8 items = items[1:]

items = [93]

6 while len([93]) > 0: --- True

7 res = '35,92,27' + ',' + str(items[0])

res = '35,92,27,93'

8 items = items[1:]

items = []

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

9 return '35,92,27,93'

Вариант: 2-1-42

1. gcd(51, 15) = 3

1 def gcd(x=51, y=15)

2 if 51 < 0: --- False

4 if 15 < 0: --- False

6 if 51 == 0: --- False

8 while 15 != 0: --- True

9 rem = 51 % 15

rem = 6

10 x = 15

11 y = 6

8 while 6 != 0: --- True

9 rem = 15 % 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(-48, 0) = 48

1 def gcd(x=-48, y=0)

2 if -48 < 0: --- True

3 x = --48

x = 48

4 if 0 < 0: --- False

6 if 48 == 0: --- False

8 while 0 != 0: --- False

12 return 48

3. hex(215) = 'D7'

3 def hex(number=215)

4 if 215 == 0: --- False

6 res = ''

7 while 215 > 0: --- True

8 digit = 215 % 16

digit = 7

9 res = DIGITS[7] + ''

res = '7'

10 number = 215 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + '7'

res = 'D7'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'D7'

4. square\_equal(7, 63, -70) = [-10.0, 1.0]

3 def square\_equal(a=7, b=63, c=-70)

4 if 7 != 0: --- True

5 D = 63\*63 - 4\*7\*-70

D = 5929

6 if 5929 > 0: --- True

7 x1 = (-63 - sqrt(5929)) / (2\*7)

x1 = -10.0

8 x2 = (-63 + sqrt(5929)) / (2\*7)

x2 = 1.0

9 return [-10.0, 1.0]

5. square\_equal(-69, 0, -64) = []

3 def square\_equal(a=-69, b=0, c=-64)

4 if -69 != 0: --- True

5 D = 0\*0 - 4\*-69\*-64

D = -17664

6 if -17664 > 0: --- False

10 elif -17664 == 0: --- False

12 else:

13 return []

6. findmax([-95, -75, -70, -7]) = -7

1 def findmax(items=[-95, -75, -70, -7])

2 if len([-95, -75, -70, -7]) == 0: --- False

4 m = items[0]

m = -95

5 i = 1

6 while 1 < len([-95, -75, -70, -7]): --- True

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

8 m = items[1]

m = -75

9 i = 1 + 1

i = 2

6 while 2 < len([-95, -75, -70, -7]): --- True

7 if -75 < items[2]: --- True

8 m = items[2]

m = -70

9 i = 2 + 1

i = 3

6 while 3 < len([-95, -75, -70, -7]): --- True

7 if -70 < items[3]: --- True

8 m = items[3]

m = -7

9 i = 3 + 1

i = 4

6 while 4 < len([-95, -75, -70, -7]): --- False

10 return -7

7. unique([67, 67, 75]) = [67, 75]

1 def unique(items=[67, 67, 75])

2 res = []

3 i = 0

4 while 0 < len([67, 67, 75]): --- 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, 75]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([67, 67, 75]): --- True

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

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

res = [67, 75]

7 i = 2 + 1

i = 3

4 while 3 < len([67, 67, 75]): --- False

8 return [67, 75]

8. join(':', [77, 5, 24]) = '77:5:24'

1 def join(sep=:, items=[77, 5, 24])

2 res = ''

3 if len([77, 5, 24]) > 0: --- True

4 res = str(items[0])

res = '77'

5 items = items[1:]

items = [5, 24]

6 while len([5, 24]) > 0: --- True

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

res = '77:5'

8 items = items[1:]

items = [24]

6 while len([24]) > 0: --- True

7 res = '77:5' + ':' + str(items[0])

res = '77:5:24'

8 items = items[1:]

items = []

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

9 return '77:5:24'

Вариант: 2-1-43

1. gcd(9, 93) = 3

1 def gcd(x=9, y=93)

2 if 9 < 0: --- False

4 if 93 < 0: --- False

6 if 9 == 0: --- False

8 while 93 != 0: --- True

9 rem = 9 % 93

rem = 9

10 x = 93

11 y = 9

8 while 9 != 0: --- True

9 rem = 93 % 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(0, 49) = 49

1 def gcd(x=0, y=49)

2 if 0 < 0: --- False

4 if 49 < 0: --- False

6 if 0 == 0: --- True

7 return 49

3. hex(216) = 'D8'

3 def hex(number=216)

4 if 216 == 0: --- False

6 res = ''

7 while 216 > 0: --- True

8 digit = 216 % 16

digit = 8

9 res = DIGITS[8] + ''

res = '8'

10 number = 216 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + '8'

res = 'D8'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'D8'

4. square\_equal(52, -91, 39) = [0.75, 1.0]

3 def square\_equal(a=52, b=-91, c=39)

4 if 52 != 0: --- True

5 D = -91\*-91 - 4\*52\*39

D = 169

6 if 169 > 0: --- True

7 x1 = (--91 - sqrt(169)) / (2\*52)

x1 = 0.75

8 x2 = (--91 + sqrt(169)) / (2\*52)

x2 = 1.0

9 return [0.75, 1.0]

5. square\_equal(-36, 71, -49) = []

3 def square\_equal(a=-36, b=71, c=-49)

4 if -36 != 0: --- True

5 D = 71\*71 - 4\*-36\*-49

D = -2015

6 if -2015 > 0: --- False

10 elif -2015 == 0: --- False

12 else:

13 return []

6. findmax([18, 72, -5, -31, -61, 20]) = 72

1 def findmax(items=[18, 72, -5, -31, -61, 20])

2 if len([18, 72, -5, -31, -61, 20]) == 0: --- False

4 m = items[0]

m = 18

5 i = 1

6 while 1 < len([18, 72, -5, -31, -61, 20]): --- True

7 if 18 < items[1]: --- True

8 m = items[1]

m = 72

9 i = 1 + 1

i = 2

6 while 2 < len([18, 72, -5, -31, -61, 20]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([18, 72, -5, -31, -61, 20]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([18, 72, -5, -31, -61, 20]): --- True

7 if 72 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([18, 72, -5, -31, -61, 20]): --- True

7 if 72 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([18, 72, -5, -31, -61, 20]): --- False

10 return 72

7. unique([95, 43, 95, 95]) = [95, 43, 95]

1 def unique(items=[95, 43, 95, 95])

2 res = []

3 i = 0

4 while 0 < len([95, 43, 95, 95]): --- True

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

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

res = [95]

7 i = 0 + 1

i = 1

4 while 1 < len([95, 43, 95, 95]): --- True

5 if len([95]) == 0 or res[-1] != items[1]: --- True

6 res = [95] + [items[1]]

res = [95, 43]

7 i = 1 + 1

i = 2

4 while 2 < len([95, 43, 95, 95]): --- True

5 if len([95, 43]) == 0 or res[-1] != items[2]: --- True

6 res = [95, 43] + [items[2]]

res = [95, 43, 95]

7 i = 2 + 1

i = 3

4 while 3 < len([95, 43, 95, 95]): --- True

5 if len([95, 43, 95]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([95, 43, 95, 95]): --- False

8 return [95, 43, 95]

8. join(':', [78, 91, 74, 76]) = '78:91:74:76'

1 def join(sep=:, items=[78, 91, 74, 76])

2 res = ''

3 if len([78, 91, 74, 76]) > 0: --- True

4 res = str(items[0])

res = '78'

5 items = items[1:]

items = [91, 74, 76]

6 while len([91, 74, 76]) > 0: --- True

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

res = '78:91'

8 items = items[1:]

items = [74, 76]

6 while len([74, 76]) > 0: --- True

7 res = '78:91' + ':' + str(items[0])

res = '78:91:74'

8 items = items[1:]

items = [76]

6 while len([76]) > 0: --- True

7 res = '78:91:74' + ':' + str(items[0])

res = '78:91:74:76'

8 items = items[1:]

items = []

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

9 return '78:91:74:76'

Вариант: 2-1-44

1. gcd(-10, -30) = 10

1 def gcd(x=-10, y=-30)

2 if -10 < 0: --- True

3 x = --10

x = 10

4 if -30 < 0: --- True

5 y = --30

y = 30

6 if 10 == 0: --- False

8 while 30 != 0: --- True

9 rem = 10 % 30

rem = 10

10 x = 30

11 y = 10

8 while 10 != 0: --- True

9 rem = 30 % 10

rem = 0

10 x = 10

11 y = 0

8 while 0 != 0: --- False

12 return 10

2. gcd(46, 0) = 46

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

2 if 46 < 0: --- False

4 if 0 < 0: --- False

6 if 46 == 0: --- False

8 while 0 != 0: --- False

12 return 46

3. hex(239) = 'EF'

3 def hex(number=239)

4 if 239 == 0: --- False

6 res = ''

7 while 239 > 0: --- True

8 digit = 239 % 16

digit = 15

9 res = DIGITS[15] + ''

res = 'F'

10 number = 239 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + 'F'

res = 'EF'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'EF'

4. square\_equal(4, -45, 0) = [0.0, 11.25]

3 def square\_equal(a=4, b=-45, c=0)

4 if 4 != 0: --- True

5 D = -45\*-45 - 4\*4\*0

D = 2025

6 if 2025 > 0: --- True

7 x1 = (--45 - sqrt(2025)) / (2\*4)

x1 = 0.0

8 x2 = (--45 + sqrt(2025)) / (2\*4)

x2 = 11.25

9 return [0.0, 11.25]

5. square\_equal(-58, 30, -86) = []

3 def square\_equal(a=-58, b=30, c=-86)

4 if -58 != 0: --- True

5 D = 30\*30 - 4\*-58\*-86

D = -19052

6 if -19052 > 0: --- False

10 elif -19052 == 0: --- False

12 else:

13 return []

6. findmax([9, 89, -23, -45]) = 89

1 def findmax(items=[9, 89, -23, -45])

2 if len([9, 89, -23, -45]) == 0: --- False

4 m = items[0]

m = 9

5 i = 1

6 while 1 < len([9, 89, -23, -45]): --- True

7 if 9 < items[1]: --- True

8 m = items[1]

m = 89

9 i = 1 + 1

i = 2

6 while 2 < len([9, 89, -23, -45]): --- True

7 if 89 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([9, 89, -23, -45]): --- True

7 if 89 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([9, 89, -23, -45]): --- False

10 return 89

7. unique([-68, 12, -41, -41]) = [-68, 12, -41]

1 def unique(items=[-68, 12, -41, -41])

2 res = []

3 i = 0

4 while 0 < len([-68, 12, -41, -41]): --- 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, 12, -41, -41]): --- True

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

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

res = [-68, 12]

7 i = 1 + 1

i = 2

4 while 2 < len([-68, 12, -41, -41]): --- True

5 if len([-68, 12]) == 0 or res[-1] != items[2]: --- True

6 res = [-68, 12] + [items[2]]

res = [-68, 12, -41]

7 i = 2 + 1

i = 3

4 while 3 < len([-68, 12, -41, -41]): --- True

5 if len([-68, 12, -41]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([-68, 12, -41, -41]): --- False

8 return [-68, 12, -41]

8. join(':', [33, 38, 71]) = '33:38:71'

1 def join(sep=:, items=[33, 38, 71])

2 res = ''

3 if len([33, 38, 71]) > 0: --- True

4 res = str(items[0])

res = '33'

5 items = items[1:]

items = [38, 71]

6 while len([38, 71]) > 0: --- True

7 res = '33' + ':' + str(items[0])

res = '33:38'

8 items = items[1:]

items = [71]

6 while len([71]) > 0: --- True

7 res = '33:38' + ':' + str(items[0])

res = '33:38:71'

8 items = items[1:]

items = []

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

9 return '33:38:71'

Вариант: 2-1-45

1. gcd(56, 44) = 4

1 def gcd(x=56, y=44)

2 if 56 < 0: --- False

4 if 44 < 0: --- False

6 if 56 == 0: --- False

8 while 44 != 0: --- True

9 rem = 56 % 44

rem = 12

10 x = 44

11 y = 12

8 while 12 != 0: --- True

9 rem = 44 % 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(-37, 0) = 37

1 def gcd(x=-37, y=0)

2 if -37 < 0: --- True

3 x = --37

x = 37

4 if 0 < 0: --- False

6 if 37 == 0: --- False

8 while 0 != 0: --- False

12 return 37

3. hex(191) = 'BF'

3 def hex(number=191)

4 if 191 == 0: --- False

6 res = ''

7 while 191 > 0: --- True

8 digit = 191 % 16

digit = 15

9 res = DIGITS[15] + ''

res = 'F'

10 number = 191 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + 'F'

res = 'BF'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'BF'

4. square\_equal(60, -18, -6) = [-0.2, 0.5]

3 def square\_equal(a=60, b=-18, c=-6)

4 if 60 != 0: --- True

5 D = -18\*-18 - 4\*60\*-6

D = 1764

6 if 1764 > 0: --- True

7 x1 = (--18 - sqrt(1764)) / (2\*60)

x1 = -0.2

8 x2 = (--18 + sqrt(1764)) / (2\*60)

x2 = 0.5

9 return [-0.2, 0.5]

5. square\_equal(93, 95, 84) = []

3 def square\_equal(a=93, b=95, c=84)

4 if 93 != 0: --- True

5 D = 95\*95 - 4\*93\*84

D = -22223

6 if -22223 > 0: --- False

10 elif -22223 == 0: --- False

12 else:

13 return []

6. findmax([-41, -10, -87, -92]) = -10

1 def findmax(items=[-41, -10, -87, -92])

2 if len([-41, -10, -87, -92]) == 0: --- False

4 m = items[0]

m = -41

5 i = 1

6 while 1 < len([-41, -10, -87, -92]): --- True

7 if -41 < items[1]: --- True

8 m = items[1]

m = -10

9 i = 1 + 1

i = 2

6 while 2 < len([-41, -10, -87, -92]): --- True

7 if -10 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-41, -10, -87, -92]): --- True

7 if -10 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-41, -10, -87, -92]): --- False

10 return -10

7. unique([-95, -95, -9, -93]) = [-95, -9, -93]

1 def unique(items=[-95, -95, -9, -93])

2 res = []

3 i = 0

4 while 0 < len([-95, -95, -9, -93]): --- True

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

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

res = [-95]

7 i = 0 + 1

i = 1

4 while 1 < len([-95, -95, -9, -93]): --- True

5 if len([-95]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-95, -95, -9, -93]): --- True

5 if len([-95]) == 0 or res[-1] != items[2]: --- True

6 res = [-95] + [items[2]]

res = [-95, -9]

7 i = 2 + 1

i = 3

4 while 3 < len([-95, -95, -9, -93]): --- True

5 if len([-95, -9]) == 0 or res[-1] != items[3]: --- True

6 res = [-95, -9] + [items[3]]

res = [-95, -9, -93]

7 i = 3 + 1

i = 4

4 while 4 < len([-95, -95, -9, -93]): --- False

8 return [-95, -9, -93]

8. join(':', [97, 58, 84]) = '97:58:84'

1 def join(sep=:, items=[97, 58, 84])

2 res = ''

3 if len([97, 58, 84]) > 0: --- True

4 res = str(items[0])

res = '97'

5 items = items[1:]

items = [58, 84]

6 while len([58, 84]) > 0: --- True

7 res = '97' + ':' + str(items[0])

res = '97:58'

8 items = items[1:]

items = [84]

6 while len([84]) > 0: --- True

7 res = '97:58' + ':' + str(items[0])

res = '97:58:84'

8 items = items[1:]

items = []

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

9 return '97:58:84'

Вариант: 2-1-46

1. gcd(-85, 51) = 17

1 def gcd(x=-85, y=51)

2 if -85 < 0: --- True

3 x = --85

x = 85

4 if 51 < 0: --- False

6 if 85 == 0: --- False

8 while 51 != 0: --- True

9 rem = 85 % 51

rem = 34

10 x = 51

11 y = 34

8 while 34 != 0: --- True

9 rem = 51 % 34

rem = 17

10 x = 34

11 y = 17

8 while 17 != 0: --- True

9 rem = 34 % 17

rem = 0

10 x = 17

11 y = 0

8 while 0 != 0: --- False

12 return 17

2. gcd(47, 0) = 47

1 def gcd(x=47, y=0)

2 if 47 < 0: --- False

4 if 0 < 0: --- False

6 if 47 == 0: --- False

8 while 0 != 0: --- False

12 return 47

3. hex(235) = 'EB'

3 def hex(number=235)

4 if 235 == 0: --- False

6 res = ''

7 while 235 > 0: --- True

8 digit = 235 % 16

digit = 11

9 res = DIGITS[11] + ''

res = 'B'

10 number = 235 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + 'B'

res = 'EB'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'EB'

4. square\_equal(-4, 42, -54) = [9.0, 1.5]

3 def square\_equal(a=-4, b=42, c=-54)

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

5 D = 42\*42 - 4\*-4\*-54

D = 900

6 if 900 > 0: --- True

7 x1 = (-42 - sqrt(900)) / (2\*-4)

x1 = 9.0

8 x2 = (-42 + sqrt(900)) / (2\*-4)

x2 = 1.5

9 return [9.0, 1.5]

5. square\_equal(-34, -36, -100) = []

3 def square\_equal(a=-34, b=-36, c=-100)

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

5 D = -36\*-36 - 4\*-34\*-100

D = -12304

6 if -12304 > 0: --- False

10 elif -12304 == 0: --- False

12 else:

13 return []

6. findmax([-38, -36, -18, 8]) = 8

1 def findmax(items=[-38, -36, -18, 8])

2 if len([-38, -36, -18, 8]) == 0: --- False

4 m = items[0]

m = -38

5 i = 1

6 while 1 < len([-38, -36, -18, 8]): --- True

7 if -38 < items[1]: --- True

8 m = items[1]

m = -36

9 i = 1 + 1

i = 2

6 while 2 < len([-38, -36, -18, 8]): --- True

7 if -36 < items[2]: --- True

8 m = items[2]

m = -18

9 i = 2 + 1

i = 3

6 while 3 < len([-38, -36, -18, 8]): --- True

7 if -18 < items[3]: --- True

8 m = items[3]

m = 8

9 i = 3 + 1

i = 4

6 while 4 < len([-38, -36, -18, 8]): --- False

10 return 8

7. unique([-97, -97, 93, -97]) = [-97, 93, -97]

1 def unique(items=[-97, -97, 93, -97])

2 res = []

3 i = 0

4 while 0 < len([-97, -97, 93, -97]): --- True

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

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

res = [-97]

7 i = 0 + 1

i = 1

4 while 1 < len([-97, -97, 93, -97]): --- True

5 if len([-97]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-97, -97, 93, -97]): --- True

5 if len([-97]) == 0 or res[-1] != items[2]: --- True

6 res = [-97] + [items[2]]

res = [-97, 93]

7 i = 2 + 1

i = 3

4 while 3 < len([-97, -97, 93, -97]): --- True

5 if len([-97, 93]) == 0 or res[-1] != items[3]: --- True

6 res = [-97, 93] + [items[3]]

res = [-97, 93, -97]

7 i = 3 + 1

i = 4

4 while 4 < len([-97, -97, 93, -97]): --- False

8 return [-97, 93, -97]

8. join(':', [44, 98, 69]) = '44:98:69'

1 def join(sep=:, items=[44, 98, 69])

2 res = ''

3 if len([44, 98, 69]) > 0: --- True

4 res = str(items[0])

res = '44'

5 items = items[1:]

items = [98, 69]

6 while len([98, 69]) > 0: --- True

7 res = '44' + ':' + str(items[0])

res = '44:98'

8 items = items[1:]

items = [69]

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

7 res = '44:98' + ':' + str(items[0])

res = '44:98:69'

8 items = items[1:]

items = []

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

9 return '44:98:69'

Вариант: 2-1-47

1. gcd(-12, 80) = 4

1 def gcd(x=-12, y=80)

2 if -12 < 0: --- True

3 x = --12

x = 12

4 if 80 < 0: --- False

6 if 12 == 0: --- False

8 while 80 != 0: --- True

9 rem = 12 % 80

rem = 12

10 x = 80

11 y = 12

8 while 12 != 0: --- True

9 rem = 80 % 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(-42, 0) = 42

1 def gcd(x=-42, y=0)

2 if -42 < 0: --- True

3 x = --42

x = 42

4 if 0 < 0: --- False

6 if 42 == 0: --- False

8 while 0 != 0: --- False

12 return 42

3. hex(238) = 'EE'

3 def hex(number=238)

4 if 238 == 0: --- False

6 res = ''

7 while 238 > 0: --- True

8 digit = 238 % 16

digit = 14

9 res = DIGITS[14] + ''

res = 'E'

10 number = 238 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + 'E'

res = 'EE'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'EE'

4. square\_equal(18, 81, 63) = [-3.5, -1.0]

3 def square\_equal(a=18, b=81, c=63)

4 if 18 != 0: --- True

5 D = 81\*81 - 4\*18\*63

D = 2025

6 if 2025 > 0: --- True

7 x1 = (-81 - sqrt(2025)) / (2\*18)

x1 = -3.5

8 x2 = (-81 + sqrt(2025)) / (2\*18)

x2 = -1.0

9 return [-3.5, -1.0]

5. square\_equal(-18, 5, -19) = []

3 def square\_equal(a=-18, b=5, c=-19)

4 if -18 != 0: --- True

5 D = 5\*5 - 4\*-18\*-19

D = -1343

6 if -1343 > 0: --- False

10 elif -1343 == 0: --- False

12 else:

13 return []

6. findmax([14, -62, 47, 38]) = 47

1 def findmax(items=[14, -62, 47, 38])

2 if len([14, -62, 47, 38]) == 0: --- False

4 m = items[0]

m = 14

5 i = 1

6 while 1 < len([14, -62, 47, 38]): --- True

7 if 14 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([14, -62, 47, 38]): --- True

7 if 14 < items[2]: --- True

8 m = items[2]

m = 47

9 i = 2 + 1

i = 3

6 while 3 < len([14, -62, 47, 38]): --- True

7 if 47 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([14, -62, 47, 38]): --- False

10 return 47

7. unique([84, 84, -56, -63]) = [84, -56, -63]

1 def unique(items=[84, 84, -56, -63])

2 res = []

3 i = 0

4 while 0 < len([84, 84, -56, -63]): --- True

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

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

res = [84]

7 i = 0 + 1

i = 1

4 while 1 < len([84, 84, -56, -63]): --- True

5 if len([84]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([84, 84, -56, -63]): --- True

5 if len([84]) == 0 or res[-1] != items[2]: --- True

6 res = [84] + [items[2]]

res = [84, -56]

7 i = 2 + 1

i = 3

4 while 3 < len([84, 84, -56, -63]): --- True

5 if len([84, -56]) == 0 or res[-1] != items[3]: --- True

6 res = [84, -56] + [items[3]]

res = [84, -56, -63]

7 i = 3 + 1

i = 4

4 while 4 < len([84, 84, -56, -63]): --- False

8 return [84, -56, -63]

8. join(';', [83, 71, 15]) = '83;71;15'

1 def join(sep=;, items=[83, 71, 15])

2 res = ''

3 if len([83, 71, 15]) > 0: --- True

4 res = str(items[0])

res = '83'

5 items = items[1:]

items = [71, 15]

6 while len([71, 15]) > 0: --- True

7 res = '83' + ';' + str(items[0])

res = '83;71'

8 items = items[1:]

items = [15]

6 while len([15]) > 0: --- True

7 res = '83;71' + ';' + str(items[0])

res = '83;71;15'

8 items = items[1:]

items = []

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

9 return '83;71;15'

Вариант: 2-1-48

1. gcd(28, 98) = 14

1 def gcd(x=28, y=98)

2 if 28 < 0: --- False

4 if 98 < 0: --- False

6 if 28 == 0: --- False

8 while 98 != 0: --- True

9 rem = 28 % 98

rem = 28

10 x = 98

11 y = 28

8 while 28 != 0: --- True

9 rem = 98 % 28

rem = 14

10 x = 28

11 y = 14

8 while 14 != 0: --- True

9 rem = 28 % 14

rem = 0

10 x = 14

11 y = 0

8 while 0 != 0: --- False

12 return 14

2. gcd(-30, 0) = 30

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

2 if -30 < 0: --- True

3 x = --30

x = 30

4 if 0 < 0: --- False

6 if 30 == 0: --- False

8 while 0 != 0: --- False

12 return 30

3. hex(207) = 'CF'

3 def hex(number=207)

4 if 207 == 0: --- False

6 res = ''

7 while 207 > 0: --- True

8 digit = 207 % 16

digit = 15

9 res = DIGITS[15] + ''

res = 'F'

10 number = 207 // 16

number = 12

7 while 12 > 0: --- True

8 digit = 12 % 16

digit = 12

9 res = DIGITS[12] + 'F'

res = 'CF'

10 number = 12 // 16

number = 0

7 while 0 > 0: --- False

11 return 'CF'

4. square\_equal(-1, 48, 49) = [49.0, -1.0]

3 def square\_equal(a=-1, b=48, c=49)

4 if -1 != 0: --- True

5 D = 48\*48 - 4\*-1\*49

D = 2500

6 if 2500 > 0: --- True

7 x1 = (-48 - sqrt(2500)) / (2\*-1)

x1 = 49.0

8 x2 = (-48 + sqrt(2500)) / (2\*-1)

x2 = -1.0

9 return [49.0, -1.0]

5. square\_equal(78, 95, 85) = []

3 def square\_equal(a=78, b=95, c=85)

4 if 78 != 0: --- True

5 D = 95\*95 - 4\*78\*85

D = -17495

6 if -17495 > 0: --- False

10 elif -17495 == 0: --- False

12 else:

13 return []

6. findmax([-13, -38, -82, 65]) = 65

1 def findmax(items=[-13, -38, -82, 65])

2 if len([-13, -38, -82, 65]) == 0: --- False

4 m = items[0]

m = -13

5 i = 1

6 while 1 < len([-13, -38, -82, 65]): --- True

7 if -13 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([-13, -38, -82, 65]): --- True

7 if -13 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-13, -38, -82, 65]): --- True

7 if -13 < items[3]: --- True

8 m = items[3]

m = 65

9 i = 3 + 1

i = 4

6 while 4 < len([-13, -38, -82, 65]): --- False

10 return 65

7. unique([-32, -37, -37]) = [-32, -37]

1 def unique(items=[-32, -37, -37])

2 res = []

3 i = 0

4 while 0 < len([-32, -37, -37]): --- True

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

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

res = [-32]

7 i = 0 + 1

i = 1

4 while 1 < len([-32, -37, -37]): --- True

5 if len([-32]) == 0 or res[-1] != items[1]: --- True

6 res = [-32] + [items[1]]

res = [-32, -37]

7 i = 1 + 1

i = 2

4 while 2 < len([-32, -37, -37]): --- True

5 if len([-32, -37]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-32, -37, -37]): --- False

8 return [-32, -37]

8. join(';', [29, 45, 68]) = '29;45;68'

1 def join(sep=;, items=[29, 45, 68])

2 res = ''

3 if len([29, 45, 68]) > 0: --- True

4 res = str(items[0])

res = '29'

5 items = items[1:]

items = [45, 68]

6 while len([45, 68]) > 0: --- True

7 res = '29' + ';' + str(items[0])

res = '29;45'

8 items = items[1:]

items = [68]

6 while len([68]) > 0: --- True

7 res = '29;45' + ';' + str(items[0])

res = '29;45;68'

8 items = items[1:]

items = []

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

9 return '29;45;68'

Вариант: 2-1-49

1. gcd(-18, 78) = 6

1 def gcd(x=-18, y=78)

2 if -18 < 0: --- True

3 x = --18

x = 18

4 if 78 < 0: --- False

6 if 18 == 0: --- False

8 while 78 != 0: --- True

9 rem = 18 % 78

rem = 18

10 x = 78

11 y = 18

8 while 18 != 0: --- True

9 rem = 78 % 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(32, 0) = 32

1 def gcd(x=32, y=0)

2 if 32 < 0: --- False

4 if 0 < 0: --- False

6 if 32 == 0: --- False

8 while 0 != 0: --- False

12 return 32

3. hex(251) = 'FB'

3 def hex(number=251)

4 if 251 == 0: --- False

6 res = ''

7 while 251 > 0: --- True

8 digit = 251 % 16

digit = 11

9 res = DIGITS[11] + ''

res = 'B'

10 number = 251 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + 'B'

res = 'FB'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'FB'

4. square\_equal(0, -80, 8) = [0.1]

3 def square\_equal(a=0, b=-80, c=8)

4 if 0 != 0: --- False

14 else:

15 if -80 != 0: --- True

16 return [0.1]

5. square\_equal(54, -14, 55) = []

3 def square\_equal(a=54, b=-14, c=55)

4 if 54 != 0: --- True

5 D = -14\*-14 - 4\*54\*55

D = -11684

6 if -11684 > 0: --- False

10 elif -11684 == 0: --- False

12 else:

13 return []

6. findmax([-56, 38, -68, 30]) = 38

1 def findmax(items=[-56, 38, -68, 30])

2 if len([-56, 38, -68, 30]) == 0: --- False

4 m = items[0]

m = -56

5 i = 1

6 while 1 < len([-56, 38, -68, 30]): --- True

7 if -56 < items[1]: --- True

8 m = items[1]

m = 38

9 i = 1 + 1

i = 2

6 while 2 < len([-56, 38, -68, 30]): --- True

7 if 38 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-56, 38, -68, 30]): --- True

7 if 38 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-56, 38, -68, 30]): --- False

10 return 38

7. unique([-95, -46, -46, 13]) = [-95, -46, 13]

1 def unique(items=[-95, -46, -46, 13])

2 res = []

3 i = 0

4 while 0 < len([-95, -46, -46, 13]): --- True

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

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

res = [-95]

7 i = 0 + 1

i = 1

4 while 1 < len([-95, -46, -46, 13]): --- True

5 if len([-95]) == 0 or res[-1] != items[1]: --- True

6 res = [-95] + [items[1]]

res = [-95, -46]

7 i = 1 + 1

i = 2

4 while 2 < len([-95, -46, -46, 13]): --- True

5 if len([-95, -46]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-95, -46, -46, 13]): --- True

5 if len([-95, -46]) == 0 or res[-1] != items[3]: --- True

6 res = [-95, -46] + [items[3]]

res = [-95, -46, 13]

7 i = 3 + 1

i = 4

4 while 4 < len([-95, -46, -46, 13]): --- False

8 return [-95, -46, 13]

8. join(':', [26, 47, 69]) = '26:47:69'

1 def join(sep=:, items=[26, 47, 69])

2 res = ''

3 if len([26, 47, 69]) > 0: --- True

4 res = str(items[0])

res = '26'

5 items = items[1:]

items = [47, 69]

6 while len([47, 69]) > 0: --- True

7 res = '26' + ':' + str(items[0])

res = '26:47'

8 items = items[1:]

items = [69]

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

7 res = '26:47' + ':' + str(items[0])

res = '26:47:69'

8 items = items[1:]

items = []

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

9 return '26:47:69'

Вариант: 2-1-50

1. gcd(-48, -40) = 8

1 def gcd(x=-48, y=-40)

2 if -48 < 0: --- True

3 x = --48

x = 48

4 if -40 < 0: --- True

5 y = --40

y = 40

6 if 48 == 0: --- False

8 while 40 != 0: --- True

9 rem = 48 % 40

rem = 8

10 x = 40

11 y = 8

8 while 8 != 0: --- True

9 rem = 40 % 8

rem = 0

10 x = 8

11 y = 0

8 while 0 != 0: --- False

12 return 8

2. gcd(0, -9) = 9

1 def gcd(x=0, y=-9)

2 if 0 < 0: --- False

4 if -9 < 0: --- True

5 y = --9

y = 9

6 if 0 == 0: --- True

7 return 9

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(12, 60, 48) = [-4.0, -1.0]

3 def square\_equal(a=12, b=60, c=48)

4 if 12 != 0: --- True

5 D = 60\*60 - 4\*12\*48

D = 1296

6 if 1296 > 0: --- True

7 x1 = (-60 - sqrt(1296)) / (2\*12)

x1 = -4.0

8 x2 = (-60 + sqrt(1296)) / (2\*12)

x2 = -1.0

9 return [-4.0, -1.0]

5. square\_equal(97, 16, 95) = []

3 def square\_equal(a=97, b=16, c=95)

4 if 97 != 0: --- True

5 D = 16\*16 - 4\*97\*95

D = -36604

6 if -36604 > 0: --- False

10 elif -36604 == 0: --- False

12 else:

13 return []

6. findmax([74, 71, -88, 10, -81]) = 74

1 def findmax(items=[74, 71, -88, 10, -81])

2 if len([74, 71, -88, 10, -81]) == 0: --- False

4 m = items[0]

m = 74

5 i = 1

6 while 1 < len([74, 71, -88, 10, -81]): --- True

7 if 74 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([74, 71, -88, 10, -81]): --- True

7 if 74 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([74, 71, -88, 10, -81]): --- True

7 if 74 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([74, 71, -88, 10, -81]): --- True

7 if 74 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([74, 71, -88, 10, -81]): --- False

10 return 74

7. unique([-14, -27, -27]) = [-14, -27]

1 def unique(items=[-14, -27, -27])

2 res = []

3 i = 0

4 while 0 < len([-14, -27, -27]): --- True

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

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

res = [-14]

7 i = 0 + 1

i = 1

4 while 1 < len([-14, -27, -27]): --- True

5 if len([-14]) == 0 or res[-1] != items[1]: --- True

6 res = [-14] + [items[1]]

res = [-14, -27]

7 i = 1 + 1

i = 2

4 while 2 < len([-14, -27, -27]): --- True

5 if len([-14, -27]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-14, -27, -27]): --- False

8 return [-14, -27]

8. join(':', [5, 15, 37]) = '5:15:37'

1 def join(sep=:, items=[5, 15, 37])

2 res = ''

3 if len([5, 15, 37]) > 0: --- True

4 res = str(items[0])

res = '5'

5 items = items[1:]

items = [15, 37]

6 while len([15, 37]) > 0: --- True

7 res = '5' + ':' + str(items[0])

res = '5:15'

8 items = items[1:]

items = [37]

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

7 res = '5:15' + ':' + str(items[0])

res = '5:15:37'

8 items = items[1:]

items = []

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

9 return '5:15:37'