Вариант: 2-2-1

1. gcd(-56, -72) = 8

1 def gcd(x=-56, y=-72)

2 if -56 < 0: --- True

3 x = --56

x = 56

4 if -72 < 0: --- True

5 y = --72

y = 72

6 if 56 == 0: --- False

8 while 72 != 0: --- True

9 rem = 56 % 72

rem = 56

10 x = 72

11 y = 56

8 while 56 != 0: --- True

9 rem = 72 % 56

rem = 16

10 x = 56

11 y = 16

8 while 16 != 0: --- True

9 rem = 56 % 16

rem = 8

10 x = 16

11 y = 8

8 while 8 != 0: --- True

9 rem = 16 % 8

rem = 0

10 x = 8

11 y = 0

8 while 0 != 0: --- False

12 return 8

2. gcd(36, 0) = 36

1 def gcd(x=36, y=0)

2 if 36 < 0: --- False

4 if 0 < 0: --- False

6 if 36 == 0: --- False

8 while 0 != 0: --- False

12 return 36

3. hex(203) = 'CB'

3 def hex(number=203)

4 if 203 == 0: --- False

6 res = ''

7 while 203 > 0: --- True

8 digit = 203 % 16

digit = 11

9 res = DIGITS[11] + ''

res = 'B'

10 number = 203 // 16

number = 12

7 while 12 > 0: --- True

8 digit = 12 % 16

digit = 12

9 res = DIGITS[12] + 'B'

res = 'CB'

10 number = 12 // 16

number = 0

7 while 0 > 0: --- False

11 return 'CB'

4. square\_equal(24, 42, -12) = [-2.0, 0.25]

3 def square\_equal(a=24, b=42, c=-12)

4 if 24 != 0: --- True

5 D = 42\*42 - 4\*24\*-12

D = 2916

6 if 2916 > 0: --- True

7 x1 = (-42 - sqrt(2916)) / (2\*24)

x1 = -2.0

8 x2 = (-42 + sqrt(2916)) / (2\*24)

x2 = 0.25

9 return [-2.0, 0.25]

5. square\_equal(32, 37, 78) = []

3 def square\_equal(a=32, b=37, c=78)

4 if 32 != 0: --- True

5 D = 37\*37 - 4\*32\*78

D = -8615

6 if -8615 > 0: --- False

10 elif -8615 == 0: --- False

12 else:

13 return []

6. findmax([-11, -61, 4, -42, 94]) = 94

1 def findmax(items=[-11, -61, 4, -42, 94])

2 if len([-11, -61, 4, -42, 94]) == 0: --- False

4 m = items[0]

m = -11

5 i = 1

6 while 1 < len([-11, -61, 4, -42, 94]): --- True

7 if -11 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([-11, -61, 4, -42, 94]): --- True

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

8 m = items[2]

m = 4

9 i = 2 + 1

i = 3

6 while 3 < len([-11, -61, 4, -42, 94]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([-11, -61, 4, -42, 94]): --- True

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

8 m = items[4]

m = 94

9 i = 4 + 1

i = 5

6 while 5 < len([-11, -61, 4, -42, 94]): --- False

10 return 94

7. unique([-1, -10, -10]) = [-1, -10]

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

2 res = []

3 i = 0

4 while 0 < len([-1, -10, -10]): --- True

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

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

res = [-1]

7 i = 0 + 1

i = 1

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

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

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

res = [-1, -10]

7 i = 1 + 1

i = 2

4 while 2 < len([-1, -10, -10]): --- True

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

7 i = 2 + 1

i = 3

4 while 3 < len([-1, -10, -10]): --- False

8 return [-1, -10]

8. join(':', [81, 91, 31]) = '81:91:31'

1 def join(sep=:, items=[81, 91, 31])

2 res = ''

3 if len([81, 91, 31]) > 0: --- True

4 res = str(items[0])

res = '81'

5 items = items[1:]

items = [91, 31]

6 while len([91, 31]) > 0: --- True

7 res = '81' + ':' + str(items[0])

res = '81:91'

8 items = items[1:]

items = [31]

6 while len([31]) > 0: --- True

7 res = '81:91' + ':' + str(items[0])

res = '81:91:31'

8 items = items[1:]

items = []

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

9 return '81:91:31'

Вариант: 2-2-2

1. gcd(-65, -91) = 13

1 def gcd(x=-65, y=-91)

2 if -65 < 0: --- True

3 x = --65

x = 65

4 if -91 < 0: --- True

5 y = --91

y = 91

6 if 65 == 0: --- False

8 while 91 != 0: --- True

9 rem = 65 % 91

rem = 65

10 x = 91

11 y = 65

8 while 65 != 0: --- True

9 rem = 91 % 65

rem = 26

10 x = 65

11 y = 26

8 while 26 != 0: --- True

9 rem = 65 % 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, -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(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, 78, -40) = [-20.0, 0.5]

3 def square\_equal(a=4, b=78, c=-40)

4 if 4 != 0: --- True

5 D = 78\*78 - 4\*4\*-40

D = 6724

6 if 6724 > 0: --- True

7 x1 = (-78 - sqrt(6724)) / (2\*4)

x1 = -20.0

8 x2 = (-78 + sqrt(6724)) / (2\*4)

x2 = 0.5

9 return [-20.0, 0.5]

5. square\_equal(35, 28, 18) = []

3 def square\_equal(a=35, b=28, c=18)

4 if 35 != 0: --- True

5 D = 28\*28 - 4\*35\*18

D = -1736

6 if -1736 > 0: --- False

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

12 else:

13 return []

6. findmax([-92, -67, 98, -22, -71]) = 98

1 def findmax(items=[-92, -67, 98, -22, -71])

2 if len([-92, -67, 98, -22, -71]) == 0: --- False

4 m = items[0]

m = -92

5 i = 1

6 while 1 < len([-92, -67, 98, -22, -71]): --- True

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

8 m = items[1]

m = -67

9 i = 1 + 1

i = 2

6 while 2 < len([-92, -67, 98, -22, -71]): --- True

7 if -67 < items[2]: --- True

8 m = items[2]

m = 98

9 i = 2 + 1

i = 3

6 while 3 < len([-92, -67, 98, -22, -71]): --- True

7 if 98 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-92, -67, 98, -22, -71]): --- True

7 if 98 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-92, -67, 98, -22, -71]): --- False

10 return 98

7. unique([-64, -94, -94]) = [-64, -94]

1 def unique(items=[-64, -94, -94])

2 res = []

3 i = 0

4 while 0 < len([-64, -94, -94]): --- 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, -94, -94]): --- True

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

6 res = [-64] + [items[1]]

res = [-64, -94]

7 i = 1 + 1

i = 2

4 while 2 < len([-64, -94, -94]): --- True

5 if len([-64, -94]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-64, -94, -94]): --- False

8 return [-64, -94]

8. join(':', [34, 9, 25, 27]) = '34:9:25:27'

1 def join(sep=:, items=[34, 9, 25, 27])

2 res = ''

3 if len([34, 9, 25, 27]) > 0: --- True

4 res = str(items[0])

res = '34'

5 items = items[1:]

items = [9, 25, 27]

6 while len([9, 25, 27]) > 0: --- True

7 res = '34' + ':' + str(items[0])

res = '34:9'

8 items = items[1:]

items = [25, 27]

6 while len([25, 27]) > 0: --- True

7 res = '34:9' + ':' + str(items[0])

res = '34:9:25'

8 items = items[1:]

items = [27]

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

7 res = '34:9:25' + ':' + str(items[0])

res = '34:9:25:27'

8 items = items[1:]

items = []

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

9 return '34:9:25:27'

Вариант: 2-2-3

1. gcd(-81, -21) = 3

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

2 if -81 < 0: --- True

3 x = --81

x = 81

4 if -21 < 0: --- True

5 y = --21

y = 21

6 if 81 == 0: --- False

8 while 21 != 0: --- True

9 rem = 81 % 21

rem = 18

10 x = 21

11 y = 18

8 while 18 != 0: --- True

9 rem = 21 % 18

rem = 3

10 x = 18

11 y = 3

8 while 3 != 0: --- True

9 rem = 18 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(0, 17) = 17

1 def gcd(x=0, y=17)

2 if 0 < 0: --- False

4 if 17 < 0: --- False

6 if 0 == 0: --- True

7 return 17

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(13, 0, 0) = [0.0]

3 def square\_equal(a=13, b=0, c=0)

4 if 13 != 0: --- True

5 D = 0\*0 - 4\*13\*0

D = 0

6 if 0 > 0: --- False

10 elif 0 == 0: --- True

11 return [0.0]

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

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

4 if 67 != 0: --- True

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

D = -5427

6 if -5427 > 0: --- False

10 elif -5427 == 0: --- False

12 else:

13 return []

6. findmax([43, 13, -48, -43, 57, -57]) = 57

1 def findmax(items=[43, 13, -48, -43, 57, -57])

2 if len([43, 13, -48, -43, 57, -57]) == 0: --- False

4 m = items[0]

m = 43

5 i = 1

6 while 1 < len([43, 13, -48, -43, 57, -57]): --- True

7 if 43 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([43, 13, -48, -43, 57, -57]): --- True

7 if 43 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([43, 13, -48, -43, 57, -57]): --- True

7 if 43 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([43, 13, -48, -43, 57, -57]): --- True

7 if 43 < items[4]: --- True

8 m = items[4]

m = 57

9 i = 4 + 1

i = 5

6 while 5 < len([43, 13, -48, -43, 57, -57]): --- True

7 if 57 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([43, 13, -48, -43, 57, -57]): --- False

10 return 57

7. unique([-63, 18, 18]) = [-63, 18]

1 def unique(items=[-63, 18, 18])

2 res = []

3 i = 0

4 while 0 < len([-63, 18, 18]): --- 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, 18, 18]): --- True

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

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

res = [-63, 18]

7 i = 1 + 1

i = 2

4 while 2 < len([-63, 18, 18]): --- True

5 if len([-63, 18]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-63, 18, 18]): --- False

8 return [-63, 18]

8. join(',', [65, 37, 52]) = '65,37,52'

1 def join(sep=,, items=[65, 37, 52])

2 res = ''

3 if len([65, 37, 52]) > 0: --- True

4 res = str(items[0])

res = '65'

5 items = items[1:]

items = [37, 52]

6 while len([37, 52]) > 0: --- True

7 res = '65' + ',' + str(items[0])

res = '65,37'

8 items = items[1:]

items = [52]

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

7 res = '65,37' + ',' + str(items[0])

res = '65,37,52'

8 items = items[1:]

items = []

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

9 return '65,37,52'

Вариант: 2-2-4

1. gcd(24, 45) = 3

1 def gcd(x=24, y=45)

2 if 24 < 0: --- False

4 if 45 < 0: --- False

6 if 24 == 0: --- False

8 while 45 != 0: --- True

9 rem = 24 % 45

rem = 24

10 x = 45

11 y = 24

8 while 24 != 0: --- True

9 rem = 45 % 24

rem = 21

10 x = 24

11 y = 21

8 while 21 != 0: --- True

9 rem = 24 % 21

rem = 3

10 x = 21

11 y = 3

8 while 3 != 0: --- True

9 rem = 21 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(28, 0) = 28

1 def gcd(x=28, y=0)

2 if 28 < 0: --- False

4 if 0 < 0: --- False

6 if 28 == 0: --- False

8 while 0 != 0: --- False

12 return 28

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(-5, 9, 18) = [3.0, -1.2]

3 def square\_equal(a=-5, b=9, c=18)

4 if -5 != 0: --- True

5 D = 9\*9 - 4\*-5\*18

D = 441

6 if 441 > 0: --- True

7 x1 = (-9 - sqrt(441)) / (2\*-5)

x1 = 3.0

8 x2 = (-9 + sqrt(441)) / (2\*-5)

x2 = -1.2

9 return [3.0, -1.2]

5. square\_equal(42, -88, 89) = []

3 def square\_equal(a=42, b=-88, c=89)

4 if 42 != 0: --- True

5 D = -88\*-88 - 4\*42\*89

D = -7208

6 if -7208 > 0: --- False

10 elif -7208 == 0: --- False

12 else:

13 return []

6. findmax([83, -41, 57, 93, 24, 56]) = 93

1 def findmax(items=[83, -41, 57, 93, 24, 56])

2 if len([83, -41, 57, 93, 24, 56]) == 0: --- False

4 m = items[0]

m = 83

5 i = 1

6 while 1 < len([83, -41, 57, 93, 24, 56]): --- True

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

9 i = 1 + 1

i = 2

6 while 2 < len([83, -41, 57, 93, 24, 56]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([83, -41, 57, 93, 24, 56]): --- True

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

8 m = items[3]

m = 93

9 i = 3 + 1

i = 4

6 while 4 < len([83, -41, 57, 93, 24, 56]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([83, -41, 57, 93, 24, 56]): --- True

7 if 93 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([83, -41, 57, 93, 24, 56]): --- False

10 return 93

7. unique([-32, -32, -76, -69]) = [-32, -76, -69]

1 def unique(items=[-32, -32, -76, -69])

2 res = []

3 i = 0

4 while 0 < len([-32, -32, -76, -69]): --- 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, -32, -76, -69]): --- True

5 if len([-32]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-32, -32, -76, -69]): --- True

5 if len([-32]) == 0 or res[-1] != items[2]: --- True

6 res = [-32] + [items[2]]

res = [-32, -76]

7 i = 2 + 1

i = 3

4 while 3 < len([-32, -32, -76, -69]): --- True

5 if len([-32, -76]) == 0 or res[-1] != items[3]: --- True

6 res = [-32, -76] + [items[3]]

res = [-32, -76, -69]

7 i = 3 + 1

i = 4

4 while 4 < len([-32, -32, -76, -69]): --- False

8 return [-32, -76, -69]

8. join(';', [0, 76, 8, 9]) = '0;76;8;9'

1 def join(sep=;, items=[0, 76, 8, 9])

2 res = ''

3 if len([0, 76, 8, 9]) > 0: --- True

4 res = str(items[0])

res = '0'

5 items = items[1:]

items = [76, 8, 9]

6 while len([76, 8, 9]) > 0: --- True

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

res = '0;76'

8 items = items[1:]

items = [8, 9]

6 while len([8, 9]) > 0: --- True

7 res = '0;76' + ';' + str(items[0])

res = '0;76;8'

8 items = items[1:]

items = [9]

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

7 res = '0;76;8' + ';' + str(items[0])

res = '0;76;8;9'

8 items = items[1:]

items = []

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

9 return '0;76;8;9'

Вариант: 2-2-5

1. gcd(-12, -8) = 4

1 def gcd(x=-12, y=-8)

2 if -12 < 0: --- True

3 x = --12

x = 12

4 if -8 < 0: --- True

5 y = --8

y = 8

6 if 12 == 0: --- False

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(90, 0) = 90

1 def gcd(x=90, y=0)

2 if 90 < 0: --- False

4 if 0 < 0: --- False

6 if 90 == 0: --- False

8 while 0 != 0: --- False

12 return 90

3. hex(231) = 'E7'

3 def hex(number=231)

4 if 231 == 0: --- False

6 res = ''

7 while 231 > 0: --- True

8 digit = 231 % 16

digit = 7

9 res = DIGITS[7] + ''

res = '7'

10 number = 231 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + '7'

res = 'E7'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'E7'

4. square\_equal(12, 48, 48) = [-2.0]

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

4 if 12 != 0: --- True

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

D = 0

6 if 0 > 0: --- False

10 elif 0 == 0: --- True

11 return [-2.0]

5. square\_equal(-43, -42, -70) = []

3 def square\_equal(a=-43, b=-42, c=-70)

4 if -43 != 0: --- True

5 D = -42\*-42 - 4\*-43\*-70

D = -10276

6 if -10276 > 0: --- False

10 elif -10276 == 0: --- False

12 else:

13 return []

6. findmax([47, 70, -39, -68, 46]) = 70

1 def findmax(items=[47, 70, -39, -68, 46])

2 if len([47, 70, -39, -68, 46]) == 0: --- False

4 m = items[0]

m = 47

5 i = 1

6 while 1 < len([47, 70, -39, -68, 46]): --- True

7 if 47 < items[1]: --- True

8 m = items[1]

m = 70

9 i = 1 + 1

i = 2

6 while 2 < len([47, 70, -39, -68, 46]): --- True

7 if 70 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([47, 70, -39, -68, 46]): --- True

7 if 70 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([47, 70, -39, -68, 46]): --- True

7 if 70 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([47, 70, -39, -68, 46]): --- False

10 return 70

7. unique([96, 17, 17, -41]) = [96, 17, -41]

1 def unique(items=[96, 17, 17, -41])

2 res = []

3 i = 0

4 while 0 < len([96, 17, 17, -41]): --- True

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

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

res = [96]

7 i = 0 + 1

i = 1

4 while 1 < len([96, 17, 17, -41]): --- True

5 if len([96]) == 0 or res[-1] != items[1]: --- True

6 res = [96] + [items[1]]

res = [96, 17]

7 i = 1 + 1

i = 2

4 while 2 < len([96, 17, 17, -41]): --- True

5 if len([96, 17]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([96, 17, 17, -41]): --- True

5 if len([96, 17]) == 0 or res[-1] != items[3]: --- True

6 res = [96, 17] + [items[3]]

res = [96, 17, -41]

7 i = 3 + 1

i = 4

4 while 4 < len([96, 17, 17, -41]): --- False

8 return [96, 17, -41]

8. join('+', [41, 13, 45, 58]) = '41+13+45+58'

1 def join(sep=+, items=[41, 13, 45, 58])

2 res = ''

3 if len([41, 13, 45, 58]) > 0: --- True

4 res = str(items[0])

res = '41'

5 items = items[1:]

items = [13, 45, 58]

6 while len([13, 45, 58]) > 0: --- True

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

res = '41+13'

8 items = items[1:]

items = [45, 58]

6 while len([45, 58]) > 0: --- True

7 res = '41+13' + '+' + str(items[0])

res = '41+13+45'

8 items = items[1:]

items = [58]

6 while len([58]) > 0: --- True

7 res = '41+13+45' + '+' + str(items[0])

res = '41+13+45+58'

8 items = items[1:]

items = []

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

9 return '41+13+45+58'

Вариант: 2-2-6

1. gcd(27, -72) = 9

1 def gcd(x=27, y=-72)

2 if 27 < 0: --- False

4 if -72 < 0: --- True

5 y = --72

y = 72

6 if 27 == 0: --- False

8 while 72 != 0: --- True

9 rem = 27 % 72

rem = 27

10 x = 72

11 y = 27

8 while 27 != 0: --- True

9 rem = 72 % 27

rem = 18

10 x = 27

11 y = 18

8 while 18 != 0: --- True

9 rem = 27 % 18

rem = 9

10 x = 18

11 y = 9

8 while 9 != 0: --- True

9 rem = 18 % 9

rem = 0

10 x = 9

11 y = 0

8 while 0 != 0: --- False

12 return 9

2. gcd(-33, 0) = 33

1 def gcd(x=-33, y=0)

2 if -33 < 0: --- True

3 x = --33

x = 33

4 if 0 < 0: --- False

6 if 33 == 0: --- False

8 while 0 != 0: --- False

12 return 33

3. hex(245) = 'F5'

3 def hex(number=245)

4 if 245 == 0: --- False

6 res = ''

7 while 245 > 0: --- True

8 digit = 245 % 16

digit = 5

9 res = DIGITS[5] + ''

res = '5'

10 number = 245 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + '5'

res = 'F5'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'F5'

4. square\_equal(2, 95, -48) = [-48.0, 0.5]

3 def square\_equal(a=2, b=95, c=-48)

4 if 2 != 0: --- True

5 D = 95\*95 - 4\*2\*-48

D = 9409

6 if 9409 > 0: --- True

7 x1 = (-95 - sqrt(9409)) / (2\*2)

x1 = -48.0

8 x2 = (-95 + sqrt(9409)) / (2\*2)

x2 = 0.5

9 return [-48.0, 0.5]

5. square\_equal(31, 74, 97) = []

3 def square\_equal(a=31, b=74, c=97)

4 if 31 != 0: --- True

5 D = 74\*74 - 4\*31\*97

D = -6552

6 if -6552 > 0: --- False

10 elif -6552 == 0: --- False

12 else:

13 return []

6. findmax([-21, 99, 17, -23, 6]) = 99

1 def findmax(items=[-21, 99, 17, -23, 6])

2 if len([-21, 99, 17, -23, 6]) == 0: --- False

4 m = items[0]

m = -21

5 i = 1

6 while 1 < len([-21, 99, 17, -23, 6]): --- True

7 if -21 < items[1]: --- True

8 m = items[1]

m = 99

9 i = 1 + 1

i = 2

6 while 2 < len([-21, 99, 17, -23, 6]): --- True

7 if 99 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-21, 99, 17, -23, 6]): --- True

7 if 99 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-21, 99, 17, -23, 6]): --- True

7 if 99 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-21, 99, 17, -23, 6]): --- False

10 return 99

7. unique([76, 76, -20]) = [76, -20]

1 def unique(items=[76, 76, -20])

2 res = []

3 i = 0

4 while 0 < len([76, 76, -20]): --- True

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

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

res = [76]

7 i = 0 + 1

i = 1

4 while 1 < len([76, 76, -20]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([76, 76, -20]): --- True

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

6 res = [76] + [items[2]]

res = [76, -20]

7 i = 2 + 1

i = 3

4 while 3 < len([76, 76, -20]): --- False

8 return [76, -20]

8. join(',', [13, 47, 14]) = '13,47,14'

1 def join(sep=,, items=[13, 47, 14])

2 res = ''

3 if len([13, 47, 14]) > 0: --- True

4 res = str(items[0])

res = '13'

5 items = items[1:]

items = [47, 14]

6 while len([47, 14]) > 0: --- True

7 res = '13' + ',' + str(items[0])

res = '13,47'

8 items = items[1:]

items = [14]

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

7 res = '13,47' + ',' + str(items[0])

res = '13,47,14'

8 items = items[1:]

items = []

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

9 return '13,47,14'

Вариант: 2-2-7

1. gcd(12, 54) = 6

1 def gcd(x=12, y=54)

2 if 12 < 0: --- False

4 if 54 < 0: --- False

6 if 12 == 0: --- False

8 while 54 != 0: --- True

9 rem = 12 % 54

rem = 12

10 x = 54

11 y = 12

8 while 12 != 0: --- True

9 rem = 54 % 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, -60) = 60

1 def gcd(x=0, y=-60)

2 if 0 < 0: --- False

4 if -60 < 0: --- True

5 y = --60

y = 60

6 if 0 == 0: --- True

7 return 60

3. hex(241) = 'F1'

3 def hex(number=241)

4 if 241 == 0: --- False

6 res = ''

7 while 241 > 0: --- True

8 digit = 241 % 16

digit = 1

9 res = DIGITS[1] + ''

res = '1'

10 number = 241 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + '1'

res = 'F1'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'F1'

4. square\_equal(12, -3, -99) = [-2.75, 3.0]

3 def square\_equal(a=12, b=-3, c=-99)

4 if 12 != 0: --- True

5 D = -3\*-3 - 4\*12\*-99

D = 4761

6 if 4761 > 0: --- True

7 x1 = (--3 - sqrt(4761)) / (2\*12)

x1 = -2.75

8 x2 = (--3 + sqrt(4761)) / (2\*12)

x2 = 3.0

9 return [-2.75, 3.0]

5. square\_equal(11, -8, 72) = []

3 def square\_equal(a=11, b=-8, c=72)

4 if 11 != 0: --- True

5 D = -8\*-8 - 4\*11\*72

D = -3104

6 if -3104 > 0: --- False

10 elif -3104 == 0: --- False

12 else:

13 return []

6. findmax([-62, -41, -26, -52, 22]) = 22

1 def findmax(items=[-62, -41, -26, -52, 22])

2 if len([-62, -41, -26, -52, 22]) == 0: --- False

4 m = items[0]

m = -62

5 i = 1

6 while 1 < len([-62, -41, -26, -52, 22]): --- True

7 if -62 < items[1]: --- True

8 m = items[1]

m = -41

9 i = 1 + 1

i = 2

6 while 2 < len([-62, -41, -26, -52, 22]): --- True

7 if -41 < items[2]: --- True

8 m = items[2]

m = -26

9 i = 2 + 1

i = 3

6 while 3 < len([-62, -41, -26, -52, 22]): --- True

7 if -26 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-62, -41, -26, -52, 22]): --- True

7 if -26 < items[4]: --- True

8 m = items[4]

m = 22

9 i = 4 + 1

i = 5

6 while 5 < len([-62, -41, -26, -52, 22]): --- False

10 return 22

7. unique([76, 95, 95]) = [76, 95]

1 def unique(items=[76, 95, 95])

2 res = []

3 i = 0

4 while 0 < len([76, 95, 95]): --- True

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

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

res = [76]

7 i = 0 + 1

i = 1

4 while 1 < len([76, 95, 95]): --- True

5 if len([76]) == 0 or res[-1] != items[1]: --- True

6 res = [76] + [items[1]]

res = [76, 95]

7 i = 1 + 1

i = 2

4 while 2 < len([76, 95, 95]): --- True

5 if len([76, 95]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([76, 95, 95]): --- False

8 return [76, 95]

8. join('+', [26, 96, 32, 85]) = '26+96+32+85'

1 def join(sep=+, items=[26, 96, 32, 85])

2 res = ''

3 if len([26, 96, 32, 85]) > 0: --- True

4 res = str(items[0])

res = '26'

5 items = items[1:]

items = [96, 32, 85]

6 while len([96, 32, 85]) > 0: --- True

7 res = '26' + '+' + str(items[0])

res = '26+96'

8 items = items[1:]

items = [32, 85]

6 while len([32, 85]) > 0: --- True

7 res = '26+96' + '+' + str(items[0])

res = '26+96+32'

8 items = items[1:]

items = [85]

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

7 res = '26+96+32' + '+' + str(items[0])

res = '26+96+32+85'

8 items = items[1:]

items = []

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

9 return '26+96+32+85'

Вариант: 2-2-8

1. gcd(-100, -16) = 4

1 def gcd(x=-100, y=-16)

2 if -100 < 0: --- True

3 x = --100

x = 100

4 if -16 < 0: --- True

5 y = --16

y = 16

6 if 100 == 0: --- False

8 while 16 != 0: --- True

9 rem = 100 % 16

rem = 4

10 x = 16

11 y = 4

8 while 4 != 0: --- True

9 rem = 16 % 4

rem = 0

10 x = 4

11 y = 0

8 while 0 != 0: --- False

12 return 4

2. gcd(-60, 0) = 60

1 def gcd(x=-60, y=0)

2 if -60 < 0: --- True

3 x = --60

x = 60

4 if 0 < 0: --- False

6 if 60 == 0: --- False

8 while 0 != 0: --- False

12 return 60

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(34, -17, 0) = [0.0, 0.5]

3 def square\_equal(a=34, b=-17, c=0)

4 if 34 != 0: --- True

5 D = -17\*-17 - 4\*34\*0

D = 289

6 if 289 > 0: --- True

7 x1 = (--17 - sqrt(289)) / (2\*34)

x1 = 0.0

8 x2 = (--17 + sqrt(289)) / (2\*34)

x2 = 0.5

9 return [0.0, 0.5]

5. square\_equal(-81, 0, -88) = []

3 def square\_equal(a=-81, b=0, c=-88)

4 if -81 != 0: --- True

5 D = 0\*0 - 4\*-81\*-88

D = -28512

6 if -28512 > 0: --- False

10 elif -28512 == 0: --- False

12 else:

13 return []

6. findmax([-45, 69, 8, 8]) = 69

1 def findmax(items=[-45, 69, 8, 8])

2 if len([-45, 69, 8, 8]) == 0: --- False

4 m = items[0]

m = -45

5 i = 1

6 while 1 < len([-45, 69, 8, 8]): --- True

7 if -45 < items[1]: --- True

8 m = items[1]

m = 69

9 i = 1 + 1

i = 2

6 while 2 < len([-45, 69, 8, 8]): --- True

7 if 69 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-45, 69, 8, 8]): --- True

7 if 69 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-45, 69, 8, 8]): --- False

10 return 69

7. unique([5, 5, -92]) = [5, -92]

1 def unique(items=[5, 5, -92])

2 res = []

3 i = 0

4 while 0 < len([5, 5, -92]): --- 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, -92]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([5, 5, -92]): --- True

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

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

res = [5, -92]

7 i = 2 + 1

i = 3

4 while 3 < len([5, 5, -92]): --- False

8 return [5, -92]

8. join('+', [28, 59, 8, 98]) = '28+59+8+98'

1 def join(sep=+, items=[28, 59, 8, 98])

2 res = ''

3 if len([28, 59, 8, 98]) > 0: --- True

4 res = str(items[0])

res = '28'

5 items = items[1:]

items = [59, 8, 98]

6 while len([59, 8, 98]) > 0: --- True

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

res = '28+59'

8 items = items[1:]

items = [8, 98]

6 while len([8, 98]) > 0: --- True

7 res = '28+59' + '+' + str(items[0])

res = '28+59+8'

8 items = items[1:]

items = [98]

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

7 res = '28+59+8' + '+' + str(items[0])

res = '28+59+8+98'

8 items = items[1:]

items = []

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

9 return '28+59+8+98'

Вариант: 2-2-9

1. gcd(100, 45) = 5

1 def gcd(x=100, y=45)

2 if 100 < 0: --- False

4 if 45 < 0: --- False

6 if 100 == 0: --- False

8 while 45 != 0: --- True

9 rem = 100 % 45

rem = 10

10 x = 45

11 y = 10

8 while 10 != 0: --- True

9 rem = 45 % 10

rem = 5

10 x = 10

11 y = 5

8 while 5 != 0: --- True

9 rem = 10 % 5

rem = 0

10 x = 5

11 y = 0

8 while 0 != 0: --- False

12 return 5

2. gcd(27, 0) = 27

1 def gcd(x=27, y=0)

2 if 27 < 0: --- False

4 if 0 < 0: --- False

6 if 27 == 0: --- False

8 while 0 != 0: --- False

12 return 27

3. hex(179) = 'B3'

3 def hex(number=179)

4 if 179 == 0: --- False

6 res = ''

7 while 179 > 0: --- True

8 digit = 179 % 16

digit = 3

9 res = DIGITS[3] + ''

res = '3'

10 number = 179 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + '3'

res = 'B3'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'B3'

4. square\_equal(-8, 58, -77) = [5.5, 1.75]

3 def square\_equal(a=-8, b=58, c=-77)

4 if -8 != 0: --- True

5 D = 58\*58 - 4\*-8\*-77

D = 900

6 if 900 > 0: --- True

7 x1 = (-58 - sqrt(900)) / (2\*-8)

x1 = 5.5

8 x2 = (-58 + sqrt(900)) / (2\*-8)

x2 = 1.75

9 return [5.5, 1.75]

5. square\_equal(-36, 9, -72) = []

3 def square\_equal(a=-36, b=9, c=-72)

4 if -36 != 0: --- True

5 D = 9\*9 - 4\*-36\*-72

D = -10287

6 if -10287 > 0: --- False

10 elif -10287 == 0: --- False

12 else:

13 return []

6. findmax([-50, -31, -84, -53, 10]) = 10

1 def findmax(items=[-50, -31, -84, -53, 10])

2 if len([-50, -31, -84, -53, 10]) == 0: --- False

4 m = items[0]

m = -50

5 i = 1

6 while 1 < len([-50, -31, -84, -53, 10]): --- True

7 if -50 < items[1]: --- True

8 m = items[1]

m = -31

9 i = 1 + 1

i = 2

6 while 2 < len([-50, -31, -84, -53, 10]): --- True

7 if -31 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-50, -31, -84, -53, 10]): --- True

7 if -31 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-50, -31, -84, -53, 10]): --- True

7 if -31 < items[4]: --- True

8 m = items[4]

m = 10

9 i = 4 + 1

i = 5

6 while 5 < len([-50, -31, -84, -53, 10]): --- False

10 return 10

7. unique([-33, -33, 45]) = [-33, 45]

1 def unique(items=[-33, -33, 45])

2 res = []

3 i = 0

4 while 0 < len([-33, -33, 45]): --- True

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

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

res = [-33]

7 i = 0 + 1

i = 1

4 while 1 < len([-33, -33, 45]): --- True

5 if len([-33]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-33, -33, 45]): --- True

5 if len([-33]) == 0 or res[-1] != items[2]: --- True

6 res = [-33] + [items[2]]

res = [-33, 45]

7 i = 2 + 1

i = 3

4 while 3 < len([-33, -33, 45]): --- False

8 return [-33, 45]

8. join(';', [65, 66, 72, 13]) = '65;66;72;13'

1 def join(sep=;, items=[65, 66, 72, 13])

2 res = ''

3 if len([65, 66, 72, 13]) > 0: --- True

4 res = str(items[0])

res = '65'

5 items = items[1:]

items = [66, 72, 13]

6 while len([66, 72, 13]) > 0: --- True

7 res = '65' + ';' + str(items[0])

res = '65;66'

8 items = items[1:]

items = [72, 13]

6 while len([72, 13]) > 0: --- True

7 res = '65;66' + ';' + str(items[0])

res = '65;66;72'

8 items = items[1:]

items = [13]

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

7 res = '65;66;72' + ';' + str(items[0])

res = '65;66;72;13'

8 items = items[1:]

items = []

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

9 return '65;66;72;13'

Вариант: 2-2-10

1. gcd(56, -72) = 8

1 def gcd(x=56, y=-72)

2 if 56 < 0: --- False

4 if -72 < 0: --- True

5 y = --72

y = 72

6 if 56 == 0: --- False

8 while 72 != 0: --- True

9 rem = 56 % 72

rem = 56

10 x = 72

11 y = 56

8 while 56 != 0: --- True

9 rem = 72 % 56

rem = 16

10 x = 56

11 y = 16

8 while 16 != 0: --- True

9 rem = 56 % 16

rem = 8

10 x = 16

11 y = 8

8 while 8 != 0: --- True

9 rem = 16 % 8

rem = 0

10 x = 8

11 y = 0

8 while 0 != 0: --- False

12 return 8

2. gcd(-76, 0) = 76

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

2 if -76 < 0: --- True

3 x = --76

x = 76

4 if 0 < 0: --- False

6 if 76 == 0: --- False

8 while 0 != 0: --- False

12 return 76

3. hex(246) = 'F6'

3 def hex(number=246)

4 if 246 == 0: --- False

6 res = ''

7 while 246 > 0: --- True

8 digit = 246 % 16

digit = 6

9 res = DIGITS[6] + ''

res = '6'

10 number = 246 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + '6'

res = 'F6'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'F6'

4. square\_equal(1, 58, -59) = [-59.0, 1.0]

3 def square\_equal(a=1, b=58, c=-59)

4 if 1 != 0: --- True

5 D = 58\*58 - 4\*1\*-59

D = 3600

6 if 3600 > 0: --- True

7 x1 = (-58 - sqrt(3600)) / (2\*1)

x1 = -59.0

8 x2 = (-58 + sqrt(3600)) / (2\*1)

x2 = 1.0

9 return [-59.0, 1.0]

5. square\_equal(-37, -26, -10) = []

3 def square\_equal(a=-37, b=-26, c=-10)

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

5 D = -26\*-26 - 4\*-37\*-10

D = -804

6 if -804 > 0: --- False

10 elif -804 == 0: --- False

12 else:

13 return []

6. findmax([54, 83, 51, -60, 14]) = 83

1 def findmax(items=[54, 83, 51, -60, 14])

2 if len([54, 83, 51, -60, 14]) == 0: --- False

4 m = items[0]

m = 54

5 i = 1

6 while 1 < len([54, 83, 51, -60, 14]): --- True

7 if 54 < items[1]: --- True

8 m = items[1]

m = 83

9 i = 1 + 1

i = 2

6 while 2 < len([54, 83, 51, -60, 14]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([54, 83, 51, -60, 14]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([54, 83, 51, -60, 14]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([54, 83, 51, -60, 14]): --- False

10 return 83

7. unique([-90, -90, 88, 30]) = [-90, 88, 30]

1 def unique(items=[-90, -90, 88, 30])

2 res = []

3 i = 0

4 while 0 < len([-90, -90, 88, 30]): --- 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, -90, 88, 30]): --- True

5 if len([-90]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-90, -90, 88, 30]): --- True

5 if len([-90]) == 0 or res[-1] != items[2]: --- True

6 res = [-90] + [items[2]]

res = [-90, 88]

7 i = 2 + 1

i = 3

4 while 3 < len([-90, -90, 88, 30]): --- True

5 if len([-90, 88]) == 0 or res[-1] != items[3]: --- True

6 res = [-90, 88] + [items[3]]

res = [-90, 88, 30]

7 i = 3 + 1

i = 4

4 while 4 < len([-90, -90, 88, 30]): --- False

8 return [-90, 88, 30]

8. join(',', [54, 46, 6, 61]) = '54,46,6,61'

1 def join(sep=,, items=[54, 46, 6, 61])

2 res = ''

3 if len([54, 46, 6, 61]) > 0: --- True

4 res = str(items[0])

res = '54'

5 items = items[1:]

items = [46, 6, 61]

6 while len([46, 6, 61]) > 0: --- True

7 res = '54' + ',' + str(items[0])

res = '54,46'

8 items = items[1:]

items = [6, 61]

6 while len([6, 61]) > 0: --- True

7 res = '54,46' + ',' + str(items[0])

res = '54,46,6'

8 items = items[1:]

items = [61]

6 while len([61]) > 0: --- True

7 res = '54,46,6' + ',' + str(items[0])

res = '54,46,6,61'

8 items = items[1:]

items = []

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

9 return '54,46,6,61'

Вариант: 2-2-11

1. gcd(87, 99) = 3

1 def gcd(x=87, y=99)

2 if 87 < 0: --- False

4 if 99 < 0: --- False

6 if 87 == 0: --- False

8 while 99 != 0: --- True

9 rem = 87 % 99

rem = 87

10 x = 99

11 y = 87

8 while 87 != 0: --- True

9 rem = 99 % 87

rem = 12

10 x = 87

11 y = 12

8 while 12 != 0: --- True

9 rem = 87 % 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(0, 57) = 57

1 def gcd(x=0, y=57)

2 if 0 < 0: --- False

4 if 57 < 0: --- False

6 if 0 == 0: --- True

7 return 57

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(-34, 51, 34) = [2.0, -0.5]

3 def square\_equal(a=-34, b=51, c=34)

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

5 D = 51\*51 - 4\*-34\*34

D = 7225

6 if 7225 > 0: --- True

7 x1 = (-51 - sqrt(7225)) / (2\*-34)

x1 = 2.0

8 x2 = (-51 + sqrt(7225)) / (2\*-34)

x2 = -0.5

9 return [2.0, -0.5]

5. square\_equal(57, 38, 94) = []

3 def square\_equal(a=57, b=38, c=94)

4 if 57 != 0: --- True

5 D = 38\*38 - 4\*57\*94

D = -19988

6 if -19988 > 0: --- False

10 elif -19988 == 0: --- False

12 else:

13 return []

6. findmax([31, -45, 64, 45, -94, -23]) = 64

1 def findmax(items=[31, -45, 64, 45, -94, -23])

2 if len([31, -45, 64, 45, -94, -23]) == 0: --- False

4 m = items[0]

m = 31

5 i = 1

6 while 1 < len([31, -45, 64, 45, -94, -23]): --- True

7 if 31 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([31, -45, 64, 45, -94, -23]): --- True

7 if 31 < items[2]: --- True

8 m = items[2]

m = 64

9 i = 2 + 1

i = 3

6 while 3 < len([31, -45, 64, 45, -94, -23]): --- True

7 if 64 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([31, -45, 64, 45, -94, -23]): --- True

7 if 64 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([31, -45, 64, 45, -94, -23]): --- True

7 if 64 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([31, -45, 64, 45, -94, -23]): --- False

10 return 64

7. unique([-70, -70, -24]) = [-70, -24]

1 def unique(items=[-70, -70, -24])

2 res = []

3 i = 0

4 while 0 < len([-70, -70, -24]): --- True

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

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

res = [-70]

7 i = 0 + 1

i = 1

4 while 1 < len([-70, -70, -24]): --- True

5 if len([-70]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-70, -70, -24]): --- True

5 if len([-70]) == 0 or res[-1] != items[2]: --- True

6 res = [-70] + [items[2]]

res = [-70, -24]

7 i = 2 + 1

i = 3

4 while 3 < len([-70, -70, -24]): --- False

8 return [-70, -24]

8. join(':', [94, 29, 41, 93]) = '94:29:41:93'

1 def join(sep=:, items=[94, 29, 41, 93])

2 res = ''

3 if len([94, 29, 41, 93]) > 0: --- True

4 res = str(items[0])

res = '94'

5 items = items[1:]

items = [29, 41, 93]

6 while len([29, 41, 93]) > 0: --- True

7 res = '94' + ':' + str(items[0])

res = '94:29'

8 items = items[1:]

items = [41, 93]

6 while len([41, 93]) > 0: --- True

7 res = '94:29' + ':' + str(items[0])

res = '94:29:41'

8 items = items[1:]

items = [93]

6 while len([93]) > 0: --- True

7 res = '94:29:41' + ':' + str(items[0])

res = '94:29:41:93'

8 items = items[1:]

items = []

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

9 return '94:29:41:93'

Вариант: 2-2-12

1. gcd(88, 100) = 4

1 def gcd(x=88, y=100)

2 if 88 < 0: --- False

4 if 100 < 0: --- False

6 if 88 == 0: --- False

8 while 100 != 0: --- True

9 rem = 88 % 100

rem = 88

10 x = 100

11 y = 88

8 while 88 != 0: --- True

9 rem = 100 % 88

rem = 12

10 x = 88

11 y = 12

8 while 12 != 0: --- True

9 rem = 88 % 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(-1, 0) = 1

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

2 if -1 < 0: --- True

3 x = --1

x = 1

4 if 0 < 0: --- False

6 if 1 == 0: --- False

8 while 0 != 0: --- False

12 return 1

3. hex(219) = 'DB'

3 def hex(number=219)

4 if 219 == 0: --- False

6 res = ''

7 while 219 > 0: --- True

8 digit = 219 % 16

digit = 11

9 res = DIGITS[11] + ''

res = 'B'

10 number = 219 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + 'B'

res = 'DB'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'DB'

4. square\_equal(5, 68, -73) = [-14.6, 1.0]

3 def square\_equal(a=5, b=68, c=-73)

4 if 5 != 0: --- True

5 D = 68\*68 - 4\*5\*-73

D = 6084

6 if 6084 > 0: --- True

7 x1 = (-68 - sqrt(6084)) / (2\*5)

x1 = -14.6

8 x2 = (-68 + sqrt(6084)) / (2\*5)

x2 = 1.0

9 return [-14.6, 1.0]

5. square\_equal(-21, 52, -70) = []

3 def square\_equal(a=-21, b=52, c=-70)

4 if -21 != 0: --- True

5 D = 52\*52 - 4\*-21\*-70

D = -3176

6 if -3176 > 0: --- False

10 elif -3176 == 0: --- False

12 else:

13 return []

6. findmax([-58, -62, 30, -95]) = 30

1 def findmax(items=[-58, -62, 30, -95])

2 if len([-58, -62, 30, -95]) == 0: --- False

4 m = items[0]

m = -58

5 i = 1

6 while 1 < len([-58, -62, 30, -95]): --- True

7 if -58 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([-58, -62, 30, -95]): --- True

7 if -58 < items[2]: --- True

8 m = items[2]

m = 30

9 i = 2 + 1

i = 3

6 while 3 < len([-58, -62, 30, -95]): --- True

7 if 30 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-58, -62, 30, -95]): --- False

10 return 30

7. unique([-1, -1, 33, 2]) = [-1, 33, 2]

1 def unique(items=[-1, -1, 33, 2])

2 res = []

3 i = 0

4 while 0 < len([-1, -1, 33, 2]): --- True

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

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

res = [-1]

7 i = 0 + 1

i = 1

4 while 1 < len([-1, -1, 33, 2]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([-1, -1, 33, 2]): --- True

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

6 res = [-1] + [items[2]]

res = [-1, 33]

7 i = 2 + 1

i = 3

4 while 3 < len([-1, -1, 33, 2]): --- True

5 if len([-1, 33]) == 0 or res[-1] != items[3]: --- True

6 res = [-1, 33] + [items[3]]

res = [-1, 33, 2]

7 i = 3 + 1

i = 4

4 while 4 < len([-1, -1, 33, 2]): --- False

8 return [-1, 33, 2]

8. join(';', [58, 41, 80, 28]) = '58;41;80;28'

1 def join(sep=;, items=[58, 41, 80, 28])

2 res = ''

3 if len([58, 41, 80, 28]) > 0: --- True

4 res = str(items[0])

res = '58'

5 items = items[1:]

items = [41, 80, 28]

6 while len([41, 80, 28]) > 0: --- True

7 res = '58' + ';' + str(items[0])

res = '58;41'

8 items = items[1:]

items = [80, 28]

6 while len([80, 28]) > 0: --- True

7 res = '58;41' + ';' + str(items[0])

res = '58;41;80'

8 items = items[1:]

items = [28]

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

7 res = '58;41;80' + ';' + str(items[0])

res = '58;41;80;28'

8 items = items[1:]

items = []

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

9 return '58;41;80;28'

Вариант: 2-2-13

1. gcd(-36, -75) = 3

1 def gcd(x=-36, y=-75)

2 if -36 < 0: --- True

3 x = --36

x = 36

4 if -75 < 0: --- True

5 y = --75

y = 75

6 if 36 == 0: --- False

8 while 75 != 0: --- True

9 rem = 36 % 75

rem = 36

10 x = 75

11 y = 36

8 while 36 != 0: --- True

9 rem = 75 % 36

rem = 3

10 x = 36

11 y = 3

8 while 3 != 0: --- True

9 rem = 36 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(0, -66) = 66

1 def gcd(x=0, y=-66)

2 if 0 < 0: --- False

4 if -66 < 0: --- True

5 y = --66

y = 66

6 if 0 == 0: --- True

7 return 66

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(-2, 4, -2) = [1.0]

3 def square\_equal(a=-2, b=4, c=-2)

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

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

D = 0

6 if 0 > 0: --- False

10 elif 0 == 0: --- True

11 return [1.0]

5. square\_equal(-62, 69, -37) = []

3 def square\_equal(a=-62, b=69, c=-37)

4 if -62 != 0: --- True

5 D = 69\*69 - 4\*-62\*-37

D = -4415

6 if -4415 > 0: --- False

10 elif -4415 == 0: --- False

12 else:

13 return []

6. findmax([-42, -60, -3, -43]) = -3

1 def findmax(items=[-42, -60, -3, -43])

2 if len([-42, -60, -3, -43]) == 0: --- False

4 m = items[0]

m = -42

5 i = 1

6 while 1 < len([-42, -60, -3, -43]): --- True

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

9 i = 1 + 1

i = 2

6 while 2 < len([-42, -60, -3, -43]): --- True

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

8 m = items[2]

m = -3

9 i = 2 + 1

i = 3

6 while 3 < len([-42, -60, -3, -43]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([-42, -60, -3, -43]): --- False

10 return -3

7. unique([-80, -80, 23]) = [-80, 23]

1 def unique(items=[-80, -80, 23])

2 res = []

3 i = 0

4 while 0 < len([-80, -80, 23]): --- True

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

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

res = [-80]

7 i = 0 + 1

i = 1

4 while 1 < len([-80, -80, 23]): --- True

5 if len([-80]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-80, -80, 23]): --- True

5 if len([-80]) == 0 or res[-1] != items[2]: --- True

6 res = [-80] + [items[2]]

res = [-80, 23]

7 i = 2 + 1

i = 3

4 while 3 < len([-80, -80, 23]): --- False

8 return [-80, 23]

8. join(':', [53, 98, 63]) = '53:98:63'

1 def join(sep=:, items=[53, 98, 63])

2 res = ''

3 if len([53, 98, 63]) > 0: --- True

4 res = str(items[0])

res = '53'

5 items = items[1:]

items = [98, 63]

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

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

res = '53:98'

8 items = items[1:]

items = [63]

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

7 res = '53:98' + ':' + str(items[0])

res = '53:98:63'

8 items = items[1:]

items = []

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

9 return '53:98:63'

Вариант: 2-2-14

1. gcd(-63, -28) = 7

1 def gcd(x=-63, y=-28)

2 if -63 < 0: --- True

3 x = --63

x = 63

4 if -28 < 0: --- True

5 y = --28

y = 28

6 if 63 == 0: --- False

8 while 28 != 0: --- True

9 rem = 63 % 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(-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(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(-15, -27, 42) = [1.0, -2.8]

3 def square\_equal(a=-15, b=-27, c=42)

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

5 D = -27\*-27 - 4\*-15\*42

D = 3249

6 if 3249 > 0: --- True

7 x1 = (--27 - sqrt(3249)) / (2\*-15)

x1 = 1.0

8 x2 = (--27 + sqrt(3249)) / (2\*-15)

x2 = -2.8

9 return [1.0, -2.8]

5. square\_equal(16, -9, 92) = []

3 def square\_equal(a=16, b=-9, c=92)

4 if 16 != 0: --- True

5 D = -9\*-9 - 4\*16\*92

D = -5807

6 if -5807 > 0: --- False

10 elif -5807 == 0: --- False

12 else:

13 return []

6. findmax([-89, 77, -51, 84, 0]) = 84

1 def findmax(items=[-89, 77, -51, 84, 0])

2 if len([-89, 77, -51, 84, 0]) == 0: --- False

4 m = items[0]

m = -89

5 i = 1

6 while 1 < len([-89, 77, -51, 84, 0]): --- True

7 if -89 < items[1]: --- True

8 m = items[1]

m = 77

9 i = 1 + 1

i = 2

6 while 2 < len([-89, 77, -51, 84, 0]): --- True

7 if 77 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-89, 77, -51, 84, 0]): --- True

7 if 77 < items[3]: --- True

8 m = items[3]

m = 84

9 i = 3 + 1

i = 4

6 while 4 < len([-89, 77, -51, 84, 0]): --- True

7 if 84 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-89, 77, -51, 84, 0]): --- False

10 return 84

7. unique([96, 96, -99, 96]) = [96, -99, 96]

1 def unique(items=[96, 96, -99, 96])

2 res = []

3 i = 0

4 while 0 < len([96, 96, -99, 96]): --- True

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

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

res = [96]

7 i = 0 + 1

i = 1

4 while 1 < len([96, 96, -99, 96]): --- True

5 if len([96]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([96, 96, -99, 96]): --- True

5 if len([96]) == 0 or res[-1] != items[2]: --- True

6 res = [96] + [items[2]]

res = [96, -99]

7 i = 2 + 1

i = 3

4 while 3 < len([96, 96, -99, 96]): --- True

5 if len([96, -99]) == 0 or res[-1] != items[3]: --- True

6 res = [96, -99] + [items[3]]

res = [96, -99, 96]

7 i = 3 + 1

i = 4

4 while 4 < len([96, 96, -99, 96]): --- False

8 return [96, -99, 96]

8. join(':', [34, 83, 36]) = '34:83:36'

1 def join(sep=:, items=[34, 83, 36])

2 res = ''

3 if len([34, 83, 36]) > 0: --- True

4 res = str(items[0])

res = '34'

5 items = items[1:]

items = [83, 36]

6 while len([83, 36]) > 0: --- True

7 res = '34' + ':' + str(items[0])

res = '34:83'

8 items = items[1:]

items = [36]

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

7 res = '34:83' + ':' + str(items[0])

res = '34:83:36'

8 items = items[1:]

items = []

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

9 return '34:83:36'

Вариант: 2-2-15

1. gcd(72, -21) = 3

1 def gcd(x=72, y=-21)

2 if 72 < 0: --- False

4 if -21 < 0: --- True

5 y = --21

y = 21

6 if 72 == 0: --- False

8 while 21 != 0: --- True

9 rem = 72 % 21

rem = 9

10 x = 21

11 y = 9

8 while 9 != 0: --- True

9 rem = 21 % 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(-24, 0) = 24

1 def gcd(x=-24, y=0)

2 if -24 < 0: --- True

3 x = --24

x = 24

4 if 0 < 0: --- False

6 if 24 == 0: --- False

8 while 0 != 0: --- False

12 return 24

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, -5, -65) = [-13.0]

3 def square\_equal(a=0, b=-5, c=-65)

4 if 0 != 0: --- False

14 else:

15 if -5 != 0: --- True

16 return [-13.0]

5. square\_equal(-90, -37, -65) = []

3 def square\_equal(a=-90, b=-37, c=-65)

4 if -90 != 0: --- True

5 D = -37\*-37 - 4\*-90\*-65

D = -22031

6 if -22031 > 0: --- False

10 elif -22031 == 0: --- False

12 else:

13 return []

6. findmax([25, 10, -4, 34, 70]) = 70

1 def findmax(items=[25, 10, -4, 34, 70])

2 if len([25, 10, -4, 34, 70]) == 0: --- False

4 m = items[0]

m = 25

5 i = 1

6 while 1 < len([25, 10, -4, 34, 70]): --- True

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

9 i = 1 + 1

i = 2

6 while 2 < len([25, 10, -4, 34, 70]): --- True

7 if 25 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([25, 10, -4, 34, 70]): --- True

7 if 25 < items[3]: --- True

8 m = items[3]

m = 34

9 i = 3 + 1

i = 4

6 while 4 < len([25, 10, -4, 34, 70]): --- True

7 if 34 < items[4]: --- True

8 m = items[4]

m = 70

9 i = 4 + 1

i = 5

6 while 5 < len([25, 10, -4, 34, 70]): --- False

10 return 70

7. unique([-43, -43, 2, -73]) = [-43, 2, -73]

1 def unique(items=[-43, -43, 2, -73])

2 res = []

3 i = 0

4 while 0 < len([-43, -43, 2, -73]): --- True

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

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

res = [-43]

7 i = 0 + 1

i = 1

4 while 1 < len([-43, -43, 2, -73]): --- True

5 if len([-43]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-43, -43, 2, -73]): --- True

5 if len([-43]) == 0 or res[-1] != items[2]: --- True

6 res = [-43] + [items[2]]

res = [-43, 2]

7 i = 2 + 1

i = 3

4 while 3 < len([-43, -43, 2, -73]): --- True

5 if len([-43, 2]) == 0 or res[-1] != items[3]: --- True

6 res = [-43, 2] + [items[3]]

res = [-43, 2, -73]

7 i = 3 + 1

i = 4

4 while 4 < len([-43, -43, 2, -73]): --- False

8 return [-43, 2, -73]

8. join(';', [17, 56, 74, 7]) = '17;56;74;7'

1 def join(sep=;, items=[17, 56, 74, 7])

2 res = ''

3 if len([17, 56, 74, 7]) > 0: --- True

4 res = str(items[0])

res = '17'

5 items = items[1:]

items = [56, 74, 7]

6 while len([56, 74, 7]) > 0: --- True

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

res = '17;56'

8 items = items[1:]

items = [74, 7]

6 while len([74, 7]) > 0: --- True

7 res = '17;56' + ';' + str(items[0])

res = '17;56;74'

8 items = items[1:]

items = [7]

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

7 res = '17;56;74' + ';' + str(items[0])

res = '17;56;74;7'

8 items = items[1:]

items = []

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

9 return '17;56;74;7'

Вариант: 2-2-16

1. gcd(-70, 55) = 5

1 def gcd(x=-70, y=55)

2 if -70 < 0: --- True

3 x = --70

x = 70

4 if 55 < 0: --- False

6 if 70 == 0: --- False

8 while 55 != 0: --- True

9 rem = 70 % 55

rem = 15

10 x = 55

11 y = 15

8 while 15 != 0: --- True

9 rem = 55 % 15

rem = 10

10 x = 15

11 y = 10

8 while 10 != 0: --- True

9 rem = 15 % 10

rem = 5

10 x = 10

11 y = 5

8 while 5 != 0: --- True

9 rem = 10 % 5

rem = 0

10 x = 5

11 y = 0

8 while 0 != 0: --- False

12 return 5

2. gcd(0, 77) = 77

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

2 if 0 < 0: --- False

4 if 77 < 0: --- False

6 if 0 == 0: --- True

7 return 77

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(-38, -95, -57) = [-1.0, -1.5]

3 def square\_equal(a=-38, b=-95, c=-57)

4 if -38 != 0: --- True

5 D = -95\*-95 - 4\*-38\*-57

D = 361

6 if 361 > 0: --- True

7 x1 = (--95 - sqrt(361)) / (2\*-38)

x1 = -1.0

8 x2 = (--95 + sqrt(361)) / (2\*-38)

x2 = -1.5

9 return [-1.0, -1.5]

5. square\_equal(-90, 6, -6) = []

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

4 if -90 != 0: --- True

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

D = -2124

6 if -2124 > 0: --- False

10 elif -2124 == 0: --- False

12 else:

13 return []

6. findmax([-23, -44, 47, 41, 9, -41]) = 47

1 def findmax(items=[-23, -44, 47, 41, 9, -41])

2 if len([-23, -44, 47, 41, 9, -41]) == 0: --- False

4 m = items[0]

m = -23

5 i = 1

6 while 1 < len([-23, -44, 47, 41, 9, -41]): --- True

7 if -23 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([-23, -44, 47, 41, 9, -41]): --- True

7 if -23 < items[2]: --- True

8 m = items[2]

m = 47

9 i = 2 + 1

i = 3

6 while 3 < len([-23, -44, 47, 41, 9, -41]): --- True

7 if 47 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-23, -44, 47, 41, 9, -41]): --- True

7 if 47 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-23, -44, 47, 41, 9, -41]): --- True

7 if 47 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([-23, -44, 47, 41, 9, -41]): --- False

10 return 47

7. unique([-3, 78, 78]) = [-3, 78]

1 def unique(items=[-3, 78, 78])

2 res = []

3 i = 0

4 while 0 < len([-3, 78, 78]): --- True

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

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

res = [-3]

7 i = 0 + 1

i = 1

4 while 1 < len([-3, 78, 78]): --- True

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

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

res = [-3, 78]

7 i = 1 + 1

i = 2

4 while 2 < len([-3, 78, 78]): --- True

5 if len([-3, 78]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-3, 78, 78]): --- False

8 return [-3, 78]

8. join(';', [70, 1, 23]) = '70;1;23'

1 def join(sep=;, items=[70, 1, 23])

2 res = ''

3 if len([70, 1, 23]) > 0: --- True

4 res = str(items[0])

res = '70'

5 items = items[1:]

items = [1, 23]

6 while len([1, 23]) > 0: --- True

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

res = '70;1'

8 items = items[1:]

items = [23]

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

7 res = '70;1' + ';' + str(items[0])

res = '70;1;23'

8 items = items[1:]

items = []

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

9 return '70;1;23'

Вариант: 2-2-17

1. gcd(9, 87) = 3

1 def gcd(x=9, y=87)

2 if 9 < 0: --- False

4 if 87 < 0: --- False

6 if 9 == 0: --- False

8 while 87 != 0: --- True

9 rem = 9 % 87

rem = 9

10 x = 87

11 y = 9

8 while 9 != 0: --- True

9 rem = 87 % 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(-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(166) = 'A6'

3 def hex(number=166)

4 if 166 == 0: --- False

6 res = ''

7 while 166 > 0: --- True

8 digit = 166 % 16

digit = 6

9 res = DIGITS[6] + ''

res = '6'

10 number = 166 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + '6'

res = 'A6'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'A6'

4. square\_equal(0, -3, -33) = [-11.0]

3 def square\_equal(a=0, b=-3, c=-33)

4 if 0 != 0: --- False

14 else:

15 if -3 != 0: --- True

16 return [-11.0]

5. square\_equal(-18, -27, -36) = []

3 def square\_equal(a=-18, b=-27, c=-36)

4 if -18 != 0: --- True

5 D = -27\*-27 - 4\*-18\*-36

D = -1863

6 if -1863 > 0: --- False

10 elif -1863 == 0: --- False

12 else:

13 return []

6. findmax([-32, 10, -17, -19, -22, -4]) = 10

1 def findmax(items=[-32, 10, -17, -19, -22, -4])

2 if len([-32, 10, -17, -19, -22, -4]) == 0: --- False

4 m = items[0]

m = -32

5 i = 1

6 while 1 < len([-32, 10, -17, -19, -22, -4]): --- True

7 if -32 < items[1]: --- True

8 m = items[1]

m = 10

9 i = 1 + 1

i = 2

6 while 2 < len([-32, 10, -17, -19, -22, -4]): --- True

7 if 10 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-32, 10, -17, -19, -22, -4]): --- True

7 if 10 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-32, 10, -17, -19, -22, -4]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([-32, 10, -17, -19, -22, -4]): --- True

7 if 10 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([-32, 10, -17, -19, -22, -4]): --- False

10 return 10

7. unique([-33, -84, -84, -3]) = [-33, -84, -3]

1 def unique(items=[-33, -84, -84, -3])

2 res = []

3 i = 0

4 while 0 < len([-33, -84, -84, -3]): --- True

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

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

res = [-33]

7 i = 0 + 1

i = 1

4 while 1 < len([-33, -84, -84, -3]): --- True

5 if len([-33]) == 0 or res[-1] != items[1]: --- True

6 res = [-33] + [items[1]]

res = [-33, -84]

7 i = 1 + 1

i = 2

4 while 2 < len([-33, -84, -84, -3]): --- True

5 if len([-33, -84]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-33, -84, -84, -3]): --- True

5 if len([-33, -84]) == 0 or res[-1] != items[3]: --- True

6 res = [-33, -84] + [items[3]]

res = [-33, -84, -3]

7 i = 3 + 1

i = 4

4 while 4 < len([-33, -84, -84, -3]): --- False

8 return [-33, -84, -3]

8. join(',', [88, 23, 30]) = '88,23,30'

1 def join(sep=,, items=[88, 23, 30])

2 res = ''

3 if len([88, 23, 30]) > 0: --- True

4 res = str(items[0])

res = '88'

5 items = items[1:]

items = [23, 30]

6 while len([23, 30]) > 0: --- True

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

res = '88,23'

8 items = items[1:]

items = [30]

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

7 res = '88,23' + ',' + str(items[0])

res = '88,23,30'

8 items = items[1:]

items = []

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

9 return '88,23,30'

Вариант: 2-2-18

1. gcd(92, 28) = 4

1 def gcd(x=92, y=28)

2 if 92 < 0: --- False

4 if 28 < 0: --- False

6 if 92 == 0: --- False

8 while 28 != 0: --- True

9 rem = 92 % 28

rem = 8

10 x = 28

11 y = 8

8 while 8 != 0: --- True

9 rem = 28 % 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, 55) = 55

1 def gcd(x=0, y=55)

2 if 0 < 0: --- False

4 if 55 < 0: --- False

6 if 0 == 0: --- True

7 return 55

3. hex(197) = 'C5'

3 def hex(number=197)

4 if 197 == 0: --- False

6 res = ''

7 while 197 > 0: --- True

8 digit = 197 % 16

digit = 5

9 res = DIGITS[5] + ''

res = '5'

10 number = 197 // 16

number = 12

7 while 12 > 0: --- True

8 digit = 12 % 16

digit = 12

9 res = DIGITS[12] + '5'

res = 'C5'

10 number = 12 // 16

number = 0

7 while 0 > 0: --- False

11 return 'C5'

4. square\_equal(10, 52, 42) = [-4.2, -1.0]

3 def square\_equal(a=10, b=52, c=42)

4 if 10 != 0: --- True

5 D = 52\*52 - 4\*10\*42

D = 1024

6 if 1024 > 0: --- True

7 x1 = (-52 - sqrt(1024)) / (2\*10)

x1 = -4.2

8 x2 = (-52 + sqrt(1024)) / (2\*10)

x2 = -1.0

9 return [-4.2, -1.0]

5. square\_equal(58, 51, 69) = []

3 def square\_equal(a=58, b=51, c=69)

4 if 58 != 0: --- True

5 D = 51\*51 - 4\*58\*69

D = -13407

6 if -13407 > 0: --- False

10 elif -13407 == 0: --- False

12 else:

13 return []

6. findmax([-42, 35, 52, 68]) = 68

1 def findmax(items=[-42, 35, 52, 68])

2 if len([-42, 35, 52, 68]) == 0: --- False

4 m = items[0]

m = -42

5 i = 1

6 while 1 < len([-42, 35, 52, 68]): --- True

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

8 m = items[1]

m = 35

9 i = 1 + 1

i = 2

6 while 2 < len([-42, 35, 52, 68]): --- True

7 if 35 < items[2]: --- True

8 m = items[2]

m = 52

9 i = 2 + 1

i = 3

6 while 3 < len([-42, 35, 52, 68]): --- True

7 if 52 < items[3]: --- True

8 m = items[3]

m = 68

9 i = 3 + 1

i = 4

6 while 4 < len([-42, 35, 52, 68]): --- False

10 return 68

7. unique([-44, -44, -28, -30]) = [-44, -28, -30]

1 def unique(items=[-44, -44, -28, -30])

2 res = []

3 i = 0

4 while 0 < len([-44, -44, -28, -30]): --- 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, -28, -30]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([-44, -44, -28, -30]): --- True

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

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

res = [-44, -28]

7 i = 2 + 1

i = 3

4 while 3 < len([-44, -44, -28, -30]): --- True

5 if len([-44, -28]) == 0 or res[-1] != items[3]: --- True

6 res = [-44, -28] + [items[3]]

res = [-44, -28, -30]

7 i = 3 + 1

i = 4

4 while 4 < len([-44, -44, -28, -30]): --- False

8 return [-44, -28, -30]

8. join(';', [48, 5, 3, 33]) = '48;5;3;33'

1 def join(sep=;, items=[48, 5, 3, 33])

2 res = ''

3 if len([48, 5, 3, 33]) > 0: --- True

4 res = str(items[0])

res = '48'

5 items = items[1:]

items = [5, 3, 33]

6 while len([5, 3, 33]) > 0: --- True

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

res = '48;5'

8 items = items[1:]

items = [3, 33]

6 while len([3, 33]) > 0: --- True

7 res = '48;5' + ';' + str(items[0])

res = '48;5;3'

8 items = items[1:]

items = [33]

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

7 res = '48;5;3' + ';' + str(items[0])

res = '48;5;3;33'

8 items = items[1:]

items = []

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

9 return '48;5;3;33'

Вариант: 2-2-19

1. gcd(-93, 72) = 3

1 def gcd(x=-93, y=72)

2 if -93 < 0: --- True

3 x = --93

x = 93

4 if 72 < 0: --- False

6 if 93 == 0: --- False

8 while 72 != 0: --- True

9 rem = 93 % 72

rem = 21

10 x = 72

11 y = 21

8 while 21 != 0: --- True

9 rem = 72 % 21

rem = 9

10 x = 21

11 y = 9

8 while 9 != 0: --- True

9 rem = 21 % 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(52, 0) = 52

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

2 if 52 < 0: --- False

4 if 0 < 0: --- False

6 if 52 == 0: --- False

8 while 0 != 0: --- False

12 return 52

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(-8, -68, -32) = [-0.5, -8.0]

3 def square\_equal(a=-8, b=-68, c=-32)

4 if -8 != 0: --- True

5 D = -68\*-68 - 4\*-8\*-32

D = 3600

6 if 3600 > 0: --- True

7 x1 = (--68 - sqrt(3600)) / (2\*-8)

x1 = -0.5

8 x2 = (--68 + sqrt(3600)) / (2\*-8)

x2 = -8.0

9 return [-0.5, -8.0]

5. square\_equal(-74, -53, -58) = []

3 def square\_equal(a=-74, b=-53, c=-58)

4 if -74 != 0: --- True

5 D = -53\*-53 - 4\*-74\*-58

D = -14359

6 if -14359 > 0: --- False

10 elif -14359 == 0: --- False

12 else:

13 return []

6. findmax([-49, -72, 26, -59, -28]) = 26

1 def findmax(items=[-49, -72, 26, -59, -28])

2 if len([-49, -72, 26, -59, -28]) == 0: --- False

4 m = items[0]

m = -49

5 i = 1

6 while 1 < len([-49, -72, 26, -59, -28]): --- True

7 if -49 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([-49, -72, 26, -59, -28]): --- True

7 if -49 < items[2]: --- True

8 m = items[2]

m = 26

9 i = 2 + 1

i = 3

6 while 3 < len([-49, -72, 26, -59, -28]): --- True

7 if 26 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-49, -72, 26, -59, -28]): --- True

7 if 26 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-49, -72, 26, -59, -28]): --- False

10 return 26

7. unique([50, -21, -21, 50]) = [50, -21, 50]

1 def unique(items=[50, -21, -21, 50])

2 res = []

3 i = 0

4 while 0 < len([50, -21, -21, 50]): --- 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, -21, -21, 50]): --- True

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

6 res = [50] + [items[1]]

res = [50, -21]

7 i = 1 + 1

i = 2

4 while 2 < len([50, -21, -21, 50]): --- True

5 if len([50, -21]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([50, -21, -21, 50]): --- True

5 if len([50, -21]) == 0 or res[-1] != items[3]: --- True

6 res = [50, -21] + [items[3]]

res = [50, -21, 50]

7 i = 3 + 1

i = 4

4 while 4 < len([50, -21, -21, 50]): --- False

8 return [50, -21, 50]

8. join(';', [76, 34, 92]) = '76;34;92'

1 def join(sep=;, items=[76, 34, 92])

2 res = ''

3 if len([76, 34, 92]) > 0: --- True

4 res = str(items[0])

res = '76'

5 items = items[1:]

items = [34, 92]

6 while len([34, 92]) > 0: --- True

7 res = '76' + ';' + str(items[0])

res = '76;34'

8 items = items[1:]

items = [92]

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

7 res = '76;34' + ';' + str(items[0])

res = '76;34;92'

8 items = items[1:]

items = []

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

9 return '76;34;92'

Вариант: 2-2-20

1. gcd(14, -49) = 7

1 def gcd(x=14, y=-49)

2 if 14 < 0: --- False

4 if -49 < 0: --- True

5 y = --49

y = 49

6 if 14 == 0: --- False

8 while 49 != 0: --- True

9 rem = 14 % 49

rem = 14

10 x = 49

11 y = 14

8 while 14 != 0: --- True

9 rem = 49 % 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(-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(160) = 'A0'

3 def hex(number=160)

4 if 160 == 0: --- False

6 res = ''

7 while 160 > 0: --- True

8 digit = 160 % 16

digit = 0

9 res = DIGITS[0] + ''

res = '0'

10 number = 160 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + '0'

res = 'A0'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'A0'

4. square\_equal(0, -98, 0) = [-0.0]

3 def square\_equal(a=0, b=-98, c=0)

4 if 0 != 0: --- False

14 else:

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

16 return [-0.0]

5. square\_equal(-14, 1, -86) = []

3 def square\_equal(a=-14, b=1, c=-86)

4 if -14 != 0: --- True

5 D = 1\*1 - 4\*-14\*-86

D = -4815

6 if -4815 > 0: --- False

10 elif -4815 == 0: --- False

12 else:

13 return []

6. findmax([-72, 71, -40, 96, -44]) = 96

1 def findmax(items=[-72, 71, -40, 96, -44])

2 if len([-72, 71, -40, 96, -44]) == 0: --- False

4 m = items[0]

m = -72

5 i = 1

6 while 1 < len([-72, 71, -40, 96, -44]): --- True

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

8 m = items[1]

m = 71

9 i = 1 + 1

i = 2

6 while 2 < len([-72, 71, -40, 96, -44]): --- True

7 if 71 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-72, 71, -40, 96, -44]): --- True

7 if 71 < items[3]: --- True

8 m = items[3]

m = 96

9 i = 3 + 1

i = 4

6 while 4 < len([-72, 71, -40, 96, -44]): --- True

7 if 96 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-72, 71, -40, 96, -44]): --- False

10 return 96

7. unique([19, 19, -17, 52]) = [19, -17, 52]

1 def unique(items=[19, 19, -17, 52])

2 res = []

3 i = 0

4 while 0 < len([19, 19, -17, 52]): --- True

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

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

res = [19]

7 i = 0 + 1

i = 1

4 while 1 < len([19, 19, -17, 52]): --- True

5 if len([19]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([19, 19, -17, 52]): --- True

5 if len([19]) == 0 or res[-1] != items[2]: --- True

6 res = [19] + [items[2]]

res = [19, -17]

7 i = 2 + 1

i = 3

4 while 3 < len([19, 19, -17, 52]): --- True

5 if len([19, -17]) == 0 or res[-1] != items[3]: --- True

6 res = [19, -17] + [items[3]]

res = [19, -17, 52]

7 i = 3 + 1

i = 4

4 while 4 < len([19, 19, -17, 52]): --- False

8 return [19, -17, 52]

8. join(':', [72, 40, 25]) = '72:40:25'

1 def join(sep=:, items=[72, 40, 25])

2 res = ''

3 if len([72, 40, 25]) > 0: --- True

4 res = str(items[0])

res = '72'

5 items = items[1:]

items = [40, 25]

6 while len([40, 25]) > 0: --- True

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

res = '72:40'

8 items = items[1:]

items = [25]

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

7 res = '72:40' + ':' + str(items[0])

res = '72:40:25'

8 items = items[1:]

items = []

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

9 return '72:40:25'

Вариант: 2-2-21

1. gcd(-77, -84) = 7

1 def gcd(x=-77, y=-84)

2 if -77 < 0: --- True

3 x = --77

x = 77

4 if -84 < 0: --- True

5 y = --84

y = 84

6 if 77 == 0: --- False

8 while 84 != 0: --- True

9 rem = 77 % 84

rem = 77

10 x = 84

11 y = 77

8 while 77 != 0: --- True

9 rem = 84 % 77

rem = 7

10 x = 77

11 y = 7

8 while 7 != 0: --- True

9 rem = 77 % 7

rem = 0

10 x = 7

11 y = 0

8 while 0 != 0: --- False

12 return 7

2. gcd(0, -76) = 76

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

2 if 0 < 0: --- False

4 if -76 < 0: --- True

5 y = --76

y = 76

6 if 0 == 0: --- True

7 return 76

3. hex(163) = 'A3'

3 def hex(number=163)

4 if 163 == 0: --- False

6 res = ''

7 while 163 > 0: --- True

8 digit = 163 % 16

digit = 3

9 res = DIGITS[3] + ''

res = '3'

10 number = 163 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + '3'

res = 'A3'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'A3'

4. square\_equal(0, 20, 46) = [-2.3]

3 def square\_equal(a=0, b=20, c=46)

4 if 0 != 0: --- False

14 else:

15 if 20 != 0: --- True

16 return [-2.3]

5. square\_equal(69, -21, 14) = []

3 def square\_equal(a=69, b=-21, c=14)

4 if 69 != 0: --- True

5 D = -21\*-21 - 4\*69\*14

D = -3423

6 if -3423 > 0: --- False

10 elif -3423 == 0: --- False

12 else:

13 return []

6. findmax([35, 23, -25, 43]) = 43

1 def findmax(items=[35, 23, -25, 43])

2 if len([35, 23, -25, 43]) == 0: --- False

4 m = items[0]

m = 35

5 i = 1

6 while 1 < len([35, 23, -25, 43]): --- True

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

9 i = 1 + 1

i = 2

6 while 2 < len([35, 23, -25, 43]): --- True

7 if 35 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([35, 23, -25, 43]): --- True

7 if 35 < items[3]: --- True

8 m = items[3]

m = 43

9 i = 3 + 1

i = 4

6 while 4 < len([35, 23, -25, 43]): --- False

10 return 43

7. unique([77, 77, 44]) = [77, 44]

1 def unique(items=[77, 77, 44])

2 res = []

3 i = 0

4 while 0 < len([77, 77, 44]): --- 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, 44]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([77, 77, 44]): --- True

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

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

res = [77, 44]

7 i = 2 + 1

i = 3

4 while 3 < len([77, 77, 44]): --- False

8 return [77, 44]

8. join(';', [0, 32, 7]) = '0;32;7'

1 def join(sep=;, items=[0, 32, 7])

2 res = ''

3 if len([0, 32, 7]) > 0: --- True

4 res = str(items[0])

res = '0'

5 items = items[1:]

items = [32, 7]

6 while len([32, 7]) > 0: --- True

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

res = '0;32'

8 items = items[1:]

items = [7]

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

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

res = '0;32;7'

8 items = items[1:]

items = []

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

9 return '0;32;7'

Вариант: 2-2-22

1. gcd(-51, 45) = 3

1 def gcd(x=-51, y=45)

2 if -51 < 0: --- True

3 x = --51

x = 51

4 if 45 < 0: --- False

6 if 51 == 0: --- False

8 while 45 != 0: --- True

9 rem = 51 % 45

rem = 6

10 x = 45

11 y = 6

8 while 6 != 0: --- True

9 rem = 45 % 6

rem = 3

10 x = 6

11 y = 3

8 while 3 != 0: --- True

9 rem = 6 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(72, 0) = 72

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

2 if 72 < 0: --- False

4 if 0 < 0: --- False

6 if 72 == 0: --- False

8 while 0 != 0: --- False

12 return 72

3. hex(177) = 'B1'

3 def hex(number=177)

4 if 177 == 0: --- False

6 res = ''

7 while 177 > 0: --- True

8 digit = 177 % 16

digit = 1

9 res = DIGITS[1] + ''

res = '1'

10 number = 177 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + '1'

res = 'B1'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'B1'

4. square\_equal(8, -16, -42) = [-1.5, 3.5]

3 def square\_equal(a=8, b=-16, c=-42)

4 if 8 != 0: --- True

5 D = -16\*-16 - 4\*8\*-42

D = 1600

6 if 1600 > 0: --- True

7 x1 = (--16 - sqrt(1600)) / (2\*8)

x1 = -1.5

8 x2 = (--16 + sqrt(1600)) / (2\*8)

x2 = 3.5

9 return [-1.5, 3.5]

5. square\_equal(42, 10, 68) = []

3 def square\_equal(a=42, b=10, c=68)

4 if 42 != 0: --- True

5 D = 10\*10 - 4\*42\*68

D = -11324

6 if -11324 > 0: --- False

10 elif -11324 == 0: --- False

12 else:

13 return []

6. findmax([-24, 42, 2, -57, -85]) = 42

1 def findmax(items=[-24, 42, 2, -57, -85])

2 if len([-24, 42, 2, -57, -85]) == 0: --- False

4 m = items[0]

m = -24

5 i = 1

6 while 1 < len([-24, 42, 2, -57, -85]): --- True

7 if -24 < items[1]: --- True

8 m = items[1]

m = 42

9 i = 1 + 1

i = 2

6 while 2 < len([-24, 42, 2, -57, -85]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([-24, 42, 2, -57, -85]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([-24, 42, 2, -57, -85]): --- True

7 if 42 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-24, 42, 2, -57, -85]): --- False

10 return 42

7. unique([89, 89, 40, 89]) = [89, 40, 89]

1 def unique(items=[89, 89, 40, 89])

2 res = []

3 i = 0

4 while 0 < len([89, 89, 40, 89]): --- 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, 40, 89]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([89, 89, 40, 89]): --- True

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

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

res = [89, 40]

7 i = 2 + 1

i = 3

4 while 3 < len([89, 89, 40, 89]): --- True

5 if len([89, 40]) == 0 or res[-1] != items[3]: --- True

6 res = [89, 40] + [items[3]]

res = [89, 40, 89]

7 i = 3 + 1

i = 4

4 while 4 < len([89, 89, 40, 89]): --- False

8 return [89, 40, 89]

8. join(',', [87, 38, 67]) = '87,38,67'

1 def join(sep=,, items=[87, 38, 67])

2 res = ''

3 if len([87, 38, 67]) > 0: --- True

4 res = str(items[0])

res = '87'

5 items = items[1:]

items = [38, 67]

6 while len([38, 67]) > 0: --- True

7 res = '87' + ',' + str(items[0])

res = '87,38'

8 items = items[1:]

items = [67]

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

7 res = '87,38' + ',' + str(items[0])

res = '87,38,67'

8 items = items[1:]

items = []

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

9 return '87,38,67'

Вариант: 2-2-23

1. gcd(-32, 68) = 4

1 def gcd(x=-32, y=68)

2 if -32 < 0: --- True

3 x = --32

x = 32

4 if 68 < 0: --- False

6 if 32 == 0: --- False

8 while 68 != 0: --- True

9 rem = 32 % 68

rem = 32

10 x = 68

11 y = 32

8 while 32 != 0: --- True

9 rem = 68 % 32

rem = 4

10 x = 32

11 y = 4

8 while 4 != 0: --- True

9 rem = 32 % 4

rem = 0

10 x = 4

11 y = 0

8 while 0 != 0: --- False

12 return 4

2. gcd(-74, 0) = 74

1 def gcd(x=-74, y=0)

2 if -74 < 0: --- True

3 x = --74

x = 74

4 if 0 < 0: --- False

6 if 74 == 0: --- False

8 while 0 != 0: --- False

12 return 74

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(16, 36, -70) = [-3.5, 1.25]

3 def square\_equal(a=16, b=36, c=-70)

4 if 16 != 0: --- True

5 D = 36\*36 - 4\*16\*-70

D = 5776

6 if 5776 > 0: --- True

7 x1 = (-36 - sqrt(5776)) / (2\*16)

x1 = -3.5

8 x2 = (-36 + sqrt(5776)) / (2\*16)

x2 = 1.25

9 return [-3.5, 1.25]

5. square\_equal(19, 19, 82) = []

3 def square\_equal(a=19, b=19, c=82)

4 if 19 != 0: --- True

5 D = 19\*19 - 4\*19\*82

D = -5871

6 if -5871 > 0: --- False

10 elif -5871 == 0: --- False

12 else:

13 return []

6. findmax([23, 34, 24, 47]) = 47

1 def findmax(items=[23, 34, 24, 47])

2 if len([23, 34, 24, 47]) == 0: --- False

4 m = items[0]

m = 23

5 i = 1

6 while 1 < len([23, 34, 24, 47]): --- True

7 if 23 < items[1]: --- True

8 m = items[1]

m = 34

9 i = 1 + 1

i = 2

6 while 2 < len([23, 34, 24, 47]): --- True

7 if 34 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([23, 34, 24, 47]): --- True

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

8 m = items[3]

m = 47

9 i = 3 + 1

i = 4

6 while 4 < len([23, 34, 24, 47]): --- False

10 return 47

7. unique([67, 87, 87]) = [67, 87]

1 def unique(items=[67, 87, 87])

2 res = []

3 i = 0

4 while 0 < len([67, 87, 87]): --- 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, 87, 87]): --- True

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

6 res = [67] + [items[1]]

res = [67, 87]

7 i = 1 + 1

i = 2

4 while 2 < len([67, 87, 87]): --- True

5 if len([67, 87]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([67, 87, 87]): --- False

8 return [67, 87]

8. join('+', [11, 30, 91]) = '11+30+91'

1 def join(sep=+, items=[11, 30, 91])

2 res = ''

3 if len([11, 30, 91]) > 0: --- True

4 res = str(items[0])

res = '11'

5 items = items[1:]

items = [30, 91]

6 while len([30, 91]) > 0: --- True

7 res = '11' + '+' + str(items[0])

res = '11+30'

8 items = items[1:]

items = [91]

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

7 res = '11+30' + '+' + str(items[0])

res = '11+30+91'

8 items = items[1:]

items = []

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

9 return '11+30+91'

Вариант: 2-2-24

1. gcd(48, -72) = 24

1 def gcd(x=48, y=-72)

2 if 48 < 0: --- False

4 if -72 < 0: --- True

5 y = --72

y = 72

6 if 48 == 0: --- False

8 while 72 != 0: --- True

9 rem = 48 % 72

rem = 48

10 x = 72

11 y = 48

8 while 48 != 0: --- True

9 rem = 72 % 48

rem = 24

10 x = 48

11 y = 24

8 while 24 != 0: --- True

9 rem = 48 % 24

rem = 0

10 x = 24

11 y = 0

8 while 0 != 0: --- False

12 return 24

2. gcd(-99, 0) = 99

1 def gcd(x=-99, y=0)

2 if -99 < 0: --- True

3 x = --99

x = 99

4 if 0 < 0: --- False

6 if 99 == 0: --- False

8 while 0 != 0: --- False

12 return 99

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(25, 0, -49) = [-1.4, 1.4]

3 def square\_equal(a=25, b=0, c=-49)

4 if 25 != 0: --- True

5 D = 0\*0 - 4\*25\*-49

D = 4900

6 if 4900 > 0: --- True

7 x1 = (-0 - sqrt(4900)) / (2\*25)

x1 = -1.4

8 x2 = (-0 + sqrt(4900)) / (2\*25)

x2 = 1.4

9 return [-1.4, 1.4]

5. square\_equal(-40, 4, -69) = []

3 def square\_equal(a=-40, b=4, c=-69)

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

5 D = 4\*4 - 4\*-40\*-69

D = -11024

6 if -11024 > 0: --- False

10 elif -11024 == 0: --- False

12 else:

13 return []

6. findmax([-39, 89, 5, 41, 62, 15]) = 89

1 def findmax(items=[-39, 89, 5, 41, 62, 15])

2 if len([-39, 89, 5, 41, 62, 15]) == 0: --- False

4 m = items[0]

m = -39

5 i = 1

6 while 1 < len([-39, 89, 5, 41, 62, 15]): --- True

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

8 m = items[1]

m = 89

9 i = 1 + 1

i = 2

6 while 2 < len([-39, 89, 5, 41, 62, 15]): --- True

7 if 89 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-39, 89, 5, 41, 62, 15]): --- True

7 if 89 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-39, 89, 5, 41, 62, 15]): --- True

7 if 89 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-39, 89, 5, 41, 62, 15]): --- True

7 if 89 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([-39, 89, 5, 41, 62, 15]): --- False

10 return 89

7. unique([96, -74, 96, 96]) = [96, -74, 96]

1 def unique(items=[96, -74, 96, 96])

2 res = []

3 i = 0

4 while 0 < len([96, -74, 96, 96]): --- True

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

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

res = [96]

7 i = 0 + 1

i = 1

4 while 1 < len([96, -74, 96, 96]): --- True

5 if len([96]) == 0 or res[-1] != items[1]: --- True

6 res = [96] + [items[1]]

res = [96, -74]

7 i = 1 + 1

i = 2

4 while 2 < len([96, -74, 96, 96]): --- True

5 if len([96, -74]) == 0 or res[-1] != items[2]: --- True

6 res = [96, -74] + [items[2]]

res = [96, -74, 96]

7 i = 2 + 1

i = 3

4 while 3 < len([96, -74, 96, 96]): --- True

5 if len([96, -74, 96]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([96, -74, 96, 96]): --- False

8 return [96, -74, 96]

8. join(',', [5, 8, 2]) = '5,8,2'

1 def join(sep=,, items=[5, 8, 2])

2 res = ''

3 if len([5, 8, 2]) > 0: --- True

4 res = str(items[0])

res = '5'

5 items = items[1:]

items = [8, 2]

6 while len([8, 2]) > 0: --- True

7 res = '5' + ',' + str(items[0])

res = '5,8'

8 items = items[1:]

items = [2]

6 while len([2]) > 0: --- True

7 res = '5,8' + ',' + str(items[0])

res = '5,8,2'

8 items = items[1:]

items = []

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

9 return '5,8,2'

Вариант: 2-2-25

1. gcd(-56, 44) = 4

1 def gcd(x=-56, y=44)

2 if -56 < 0: --- True

3 x = --56

x = 56

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(-54, 0) = 54

1 def gcd(x=-54, y=0)

2 if -54 < 0: --- True

3 x = --54

x = 54

4 if 0 < 0: --- False

6 if 54 == 0: --- False

8 while 0 != 0: --- False

12 return 54

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(10, 71, -72) = [-8.0, 0.9]

3 def square\_equal(a=10, b=71, c=-72)

4 if 10 != 0: --- True

5 D = 71\*71 - 4\*10\*-72

D = 7921

6 if 7921 > 0: --- True

7 x1 = (-71 - sqrt(7921)) / (2\*10)

x1 = -8.0

8 x2 = (-71 + sqrt(7921)) / (2\*10)

x2 = 0.9

9 return [-8.0, 0.9]

5. square\_equal(66, 67, 57) = []

3 def square\_equal(a=66, b=67, c=57)

4 if 66 != 0: --- True

5 D = 67\*67 - 4\*66\*57

D = -10559

6 if -10559 > 0: --- False

10 elif -10559 == 0: --- False

12 else:

13 return []

6. findmax([-45, -34, -18, 59]) = 59

1 def findmax(items=[-45, -34, -18, 59])

2 if len([-45, -34, -18, 59]) == 0: --- False

4 m = items[0]

m = -45

5 i = 1

6 while 1 < len([-45, -34, -18, 59]): --- True

7 if -45 < items[1]: --- True

8 m = items[1]

m = -34

9 i = 1 + 1

i = 2

6 while 2 < len([-45, -34, -18, 59]): --- True

7 if -34 < items[2]: --- True

8 m = items[2]

m = -18

9 i = 2 + 1

i = 3

6 while 3 < len([-45, -34, -18, 59]): --- True

7 if -18 < items[3]: --- True

8 m = items[3]

m = 59

9 i = 3 + 1

i = 4

6 while 4 < len([-45, -34, -18, 59]): --- False

10 return 59

7. unique([44, 44, 52]) = [44, 52]

1 def unique(items=[44, 44, 52])

2 res = []

3 i = 0

4 while 0 < len([44, 44, 52]): --- 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, 52]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([44, 44, 52]): --- True

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

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

res = [44, 52]

7 i = 2 + 1

i = 3

4 while 3 < len([44, 44, 52]): --- False

8 return [44, 52]

8. join('+', [89, 58, 13, 56]) = '89+58+13+56'

1 def join(sep=+, items=[89, 58, 13, 56])

2 res = ''

3 if len([89, 58, 13, 56]) > 0: --- True

4 res = str(items[0])

res = '89'

5 items = items[1:]

items = [58, 13, 56]

6 while len([58, 13, 56]) > 0: --- True

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

res = '89+58'

8 items = items[1:]

items = [13, 56]

6 while len([13, 56]) > 0: --- True

7 res = '89+58' + '+' + str(items[0])

res = '89+58+13'

8 items = items[1:]

items = [56]

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

7 res = '89+58+13' + '+' + str(items[0])

res = '89+58+13+56'

8 items = items[1:]

items = []

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

9 return '89+58+13+56'

Вариант: 2-2-26

1. gcd(30, -99) = 3

1 def gcd(x=30, y=-99)

2 if 30 < 0: --- False

4 if -99 < 0: --- True

5 y = --99

y = 99

6 if 30 == 0: --- False

8 while 99 != 0: --- True

9 rem = 30 % 99

rem = 30

10 x = 99

11 y = 30

8 while 30 != 0: --- True

9 rem = 99 % 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(22, 0) = 22

1 def gcd(x=22, y=0)

2 if 22 < 0: --- False

4 if 0 < 0: --- False

6 if 22 == 0: --- False

8 while 0 != 0: --- False

12 return 22

3. hex(211) = 'D3'

3 def hex(number=211)

4 if 211 == 0: --- False

6 res = ''

7 while 211 > 0: --- True

8 digit = 211 % 16

digit = 3

9 res = DIGITS[3] + ''

res = '3'

10 number = 211 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + '3'

res = 'D3'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'D3'

4. square\_equal(4, 11, -38) = [-4.75, 2.0]

3 def square\_equal(a=4, b=11, c=-38)

4 if 4 != 0: --- True

5 D = 11\*11 - 4\*4\*-38

D = 729

6 if 729 > 0: --- True

7 x1 = (-11 - sqrt(729)) / (2\*4)

x1 = -4.75

8 x2 = (-11 + sqrt(729)) / (2\*4)

x2 = 2.0

9 return [-4.75, 2.0]

5. square\_equal(45, -15, 44) = []

3 def square\_equal(a=45, b=-15, c=44)

4 if 45 != 0: --- True

5 D = -15\*-15 - 4\*45\*44

D = -7695

6 if -7695 > 0: --- False

10 elif -7695 == 0: --- False

12 else:

13 return []

6. findmax([77, 2, 56, 87]) = 87

1 def findmax(items=[77, 2, 56, 87])

2 if len([77, 2, 56, 87]) == 0: --- False

4 m = items[0]

m = 77

5 i = 1

6 while 1 < len([77, 2, 56, 87]): --- True

7 if 77 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([77, 2, 56, 87]): --- True

7 if 77 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([77, 2, 56, 87]): --- True

7 if 77 < items[3]: --- True

8 m = items[3]

m = 87

9 i = 3 + 1

i = 4

6 while 4 < len([77, 2, 56, 87]): --- False

10 return 87

7. unique([90, 90, 51, -53]) = [90, 51, -53]

1 def unique(items=[90, 90, 51, -53])

2 res = []

3 i = 0

4 while 0 < len([90, 90, 51, -53]): --- 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, 90, 51, -53]): --- True

5 if len([90]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([90, 90, 51, -53]): --- True

5 if len([90]) == 0 or res[-1] != items[2]: --- True

6 res = [90] + [items[2]]

res = [90, 51]

7 i = 2 + 1

i = 3

4 while 3 < len([90, 90, 51, -53]): --- True

5 if len([90, 51]) == 0 or res[-1] != items[3]: --- True

6 res = [90, 51] + [items[3]]

res = [90, 51, -53]

7 i = 3 + 1

i = 4

4 while 4 < len([90, 90, 51, -53]): --- False

8 return [90, 51, -53]

8. join('+', [46, 24, 43, 67]) = '46+24+43+67'

1 def join(sep=+, items=[46, 24, 43, 67])

2 res = ''

3 if len([46, 24, 43, 67]) > 0: --- True

4 res = str(items[0])

res = '46'

5 items = items[1:]

items = [24, 43, 67]

6 while len([24, 43, 67]) > 0: --- True

7 res = '46' + '+' + str(items[0])

res = '46+24'

8 items = items[1:]

items = [43, 67]

6 while len([43, 67]) > 0: --- True

7 res = '46+24' + '+' + str(items[0])

res = '46+24+43'

8 items = items[1:]

items = [67]

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

7 res = '46+24+43' + '+' + str(items[0])

res = '46+24+43+67'

8 items = items[1:]

items = []

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

9 return '46+24+43+67'

Вариант: 2-2-27

1. gcd(-9, -99) = 9

1 def gcd(x=-9, y=-99)

2 if -9 < 0: --- True

3 x = --9

x = 9

4 if -99 < 0: --- True

5 y = --99

y = 99

6 if 9 == 0: --- False

8 while 99 != 0: --- True

9 rem = 9 % 99

rem = 9

10 x = 99

11 y = 9

8 while 9 != 0: --- True

9 rem = 99 % 9

rem = 0

10 x = 9

11 y = 0

8 while 0 != 0: --- False

12 return 9

2. gcd(0, -46) = 46

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

2 if 0 < 0: --- False

4 if -46 < 0: --- True

5 y = --46

y = 46

6 if 0 == 0: --- True

7 return 46

3. hex(240) = 'F0'

3 def hex(number=240)

4 if 240 == 0: --- False

6 res = ''

7 while 240 > 0: --- True

8 digit = 240 % 16

digit = 0

9 res = DIGITS[0] + ''

res = '0'

10 number = 240 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + '0'

res = 'F0'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'F0'

4. square\_equal(0, 20, -50) = [2.5]

3 def square\_equal(a=0, b=20, c=-50)

4 if 0 != 0: --- False

14 else:

15 if 20 != 0: --- True

16 return [2.5]

5. square\_equal(76, 23, 93) = []

3 def square\_equal(a=76, b=23, c=93)

4 if 76 != 0: --- True

5 D = 23\*23 - 4\*76\*93

D = -27743

6 if -27743 > 0: --- False

10 elif -27743 == 0: --- False

12 else:

13 return []

6. findmax([-24, -40, -19, 76]) = 76

1 def findmax(items=[-24, -40, -19, 76])

2 if len([-24, -40, -19, 76]) == 0: --- False

4 m = items[0]

m = -24

5 i = 1

6 while 1 < len([-24, -40, -19, 76]): --- True

7 if -24 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([-24, -40, -19, 76]): --- True

7 if -24 < items[2]: --- True

8 m = items[2]

m = -19

9 i = 2 + 1

i = 3

6 while 3 < len([-24, -40, -19, 76]): --- True

7 if -19 < items[3]: --- True

8 m = items[3]

m = 76

9 i = 3 + 1

i = 4

6 while 4 < len([-24, -40, -19, 76]): --- False

10 return 76

7. unique([-30, 80, -30, -30]) = [-30, 80, -30]

1 def unique(items=[-30, 80, -30, -30])

2 res = []

3 i = 0

4 while 0 < len([-30, 80, -30, -30]): --- True

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

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

res = [-30]

7 i = 0 + 1

i = 1

4 while 1 < len([-30, 80, -30, -30]): --- True

5 if len([-30]) == 0 or res[-1] != items[1]: --- True

6 res = [-30] + [items[1]]

res = [-30, 80]

7 i = 1 + 1

i = 2

4 while 2 < len([-30, 80, -30, -30]): --- True

5 if len([-30, 80]) == 0 or res[-1] != items[2]: --- True

6 res = [-30, 80] + [items[2]]

res = [-30, 80, -30]

7 i = 2 + 1

i = 3

4 while 3 < len([-30, 80, -30, -30]): --- True

5 if len([-30, 80, -30]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([-30, 80, -30, -30]): --- False

8 return [-30, 80, -30]

8. join(':', [76, 15, 24, 3]) = '76:15:24:3'

1 def join(sep=:, items=[76, 15, 24, 3])

2 res = ''

3 if len([76, 15, 24, 3]) > 0: --- True

4 res = str(items[0])

res = '76'

5 items = items[1:]

items = [15, 24, 3]

6 while len([15, 24, 3]) > 0: --- True

7 res = '76' + ':' + str(items[0])

res = '76:15'

8 items = items[1:]

items = [24, 3]

6 while len([24, 3]) > 0: --- True

7 res = '76:15' + ':' + str(items[0])

res = '76:15:24'

8 items = items[1:]

items = [3]

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

7 res = '76:15:24' + ':' + str(items[0])

res = '76:15:24:3'

8 items = items[1:]

items = []

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

9 return '76:15:24:3'

Вариант: 2-2-28

1. gcd(51, -54) = 3

1 def gcd(x=51, y=-54)

2 if 51 < 0: --- False

4 if -54 < 0: --- True

5 y = --54

y = 54

6 if 51 == 0: --- False

8 while 54 != 0: --- True

9 rem = 51 % 54

rem = 51

10 x = 54

11 y = 51

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, -40) = 40

1 def gcd(x=0, y=-40)

2 if 0 < 0: --- False

4 if -40 < 0: --- True

5 y = --40

y = 40

6 if 0 == 0: --- True

7 return 40

3. hex(189) = 'BD'

3 def hex(number=189)

4 if 189 == 0: --- False

6 res = ''

7 while 189 > 0: --- True

8 digit = 189 % 16

digit = 13

9 res = DIGITS[13] + ''

res = 'D'

10 number = 189 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + 'D'

res = 'BD'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'BD'

4. square\_equal(25, -77, 54) = [1.08, 2.0]

3 def square\_equal(a=25, b=-77, c=54)

4 if 25 != 0: --- True

5 D = -77\*-77 - 4\*25\*54

D = 529

6 if 529 > 0: --- True

7 x1 = (--77 - sqrt(529)) / (2\*25)

x1 = 1.08

8 x2 = (--77 + sqrt(529)) / (2\*25)

x2 = 2.0

9 return [1.08, 2.0]

5. square\_equal(-9, 30, -50) = []

3 def square\_equal(a=-9, b=30, c=-50)

4 if -9 != 0: --- True

5 D = 30\*30 - 4\*-9\*-50

D = -900

6 if -900 > 0: --- False

10 elif -900 == 0: --- False

12 else:

13 return []

6. findmax([-22, -46, 56, 67, 5]) = 67

1 def findmax(items=[-22, -46, 56, 67, 5])

2 if len([-22, -46, 56, 67, 5]) == 0: --- False

4 m = items[0]

m = -22

5 i = 1

6 while 1 < len([-22, -46, 56, 67, 5]): --- True

7 if -22 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([-22, -46, 56, 67, 5]): --- True

7 if -22 < items[2]: --- True

8 m = items[2]

m = 56

9 i = 2 + 1

i = 3

6 while 3 < len([-22, -46, 56, 67, 5]): --- True

7 if 56 < items[3]: --- True

8 m = items[3]

m = 67

9 i = 3 + 1

i = 4

6 while 4 < len([-22, -46, 56, 67, 5]): --- True

7 if 67 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-22, -46, 56, 67, 5]): --- False

10 return 67

7. unique([-95, 52, 52, -95]) = [-95, 52, -95]

1 def unique(items=[-95, 52, 52, -95])

2 res = []

3 i = 0

4 while 0 < len([-95, 52, 52, -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, 52, 52, -95]): --- True

5 if len([-95]) == 0 or res[-1] != items[1]: --- True

6 res = [-95] + [items[1]]

res = [-95, 52]

7 i = 1 + 1

i = 2

4 while 2 < len([-95, 52, 52, -95]): --- True

5 if len([-95, 52]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-95, 52, 52, -95]): --- True

5 if len([-95, 52]) == 0 or res[-1] != items[3]: --- True

6 res = [-95, 52] + [items[3]]

res = [-95, 52, -95]

7 i = 3 + 1

i = 4

4 while 4 < len([-95, 52, 52, -95]): --- False

8 return [-95, 52, -95]

8. join(',', [76, 55, 27]) = '76,55,27'

1 def join(sep=,, items=[76, 55, 27])

2 res = ''

3 if len([76, 55, 27]) > 0: --- True

4 res = str(items[0])

res = '76'

5 items = items[1:]

items = [55, 27]

6 while len([55, 27]) > 0: --- True

7 res = '76' + ',' + str(items[0])

res = '76,55'

8 items = items[1:]

items = [27]

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

7 res = '76,55' + ',' + str(items[0])

res = '76,55,27'

8 items = items[1:]

items = []

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

9 return '76,55,27'

Вариант: 2-2-29

1. gcd(-69, -46) = 23

1 def gcd(x=-69, y=-46)

2 if -69 < 0: --- True

3 x = --69

x = 69

4 if -46 < 0: --- True

5 y = --46

y = 46

6 if 69 == 0: --- False

8 while 46 != 0: --- True

9 rem = 69 % 46

rem = 23

10 x = 46

11 y = 23

8 while 23 != 0: --- True

9 rem = 46 % 23

rem = 0

10 x = 23

11 y = 0

8 while 0 != 0: --- False

12 return 23

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(175) = 'AF'

3 def hex(number=175)

4 if 175 == 0: --- False

6 res = ''

7 while 175 > 0: --- True

8 digit = 175 % 16

digit = 15

9 res = DIGITS[15] + ''

res = 'F'

10 number = 175 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + 'F'

res = 'AF'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'AF'

4. square\_equal(-2, 27, -85) = [8.5, 5.0]

3 def square\_equal(a=-2, b=27, c=-85)

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

5 D = 27\*27 - 4\*-2\*-85

D = 49

6 if 49 > 0: --- True

7 x1 = (-27 - sqrt(49)) / (2\*-2)

x1 = 8.5

8 x2 = (-27 + sqrt(49)) / (2\*-2)

x2 = 5.0

9 return [8.5, 5.0]

5. square\_equal(41, 35, 47) = []

3 def square\_equal(a=41, b=35, c=47)

4 if 41 != 0: --- True

5 D = 35\*35 - 4\*41\*47

D = -6483

6 if -6483 > 0: --- False

10 elif -6483 == 0: --- False

12 else:

13 return []

6. findmax([64, 84, 20, -7, 96]) = 96

1 def findmax(items=[64, 84, 20, -7, 96])

2 if len([64, 84, 20, -7, 96]) == 0: --- False

4 m = items[0]

m = 64

5 i = 1

6 while 1 < len([64, 84, 20, -7, 96]): --- True

7 if 64 < items[1]: --- True

8 m = items[1]

m = 84

9 i = 1 + 1

i = 2

6 while 2 < len([64, 84, 20, -7, 96]): --- True

7 if 84 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([64, 84, 20, -7, 96]): --- True

7 if 84 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([64, 84, 20, -7, 96]): --- True

7 if 84 < items[4]: --- True

8 m = items[4]

m = 96

9 i = 4 + 1

i = 5

6 while 5 < len([64, 84, 20, -7, 96]): --- False

10 return 96

7. unique([-78, -78, -64]) = [-78, -64]

1 def unique(items=[-78, -78, -64])

2 res = []

3 i = 0

4 while 0 < len([-78, -78, -64]): --- True

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

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

res = [-78]

7 i = 0 + 1

i = 1

4 while 1 < len([-78, -78, -64]): --- True

5 if len([-78]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-78, -78, -64]): --- True

5 if len([-78]) == 0 or res[-1] != items[2]: --- True

6 res = [-78] + [items[2]]

res = [-78, -64]

7 i = 2 + 1

i = 3

4 while 3 < len([-78, -78, -64]): --- False

8 return [-78, -64]

8. join(':', [75, 29, 55, 73]) = '75:29:55:73'

1 def join(sep=:, items=[75, 29, 55, 73])

2 res = ''

3 if len([75, 29, 55, 73]) > 0: --- True

4 res = str(items[0])

res = '75'

5 items = items[1:]

items = [29, 55, 73]

6 while len([29, 55, 73]) > 0: --- True

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

res = '75:29'

8 items = items[1:]

items = [55, 73]

6 while len([55, 73]) > 0: --- True

7 res = '75:29' + ':' + str(items[0])

res = '75:29:55'

8 items = items[1:]

items = [73]

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

7 res = '75:29:55' + ':' + str(items[0])

res = '75:29:55:73'

8 items = items[1:]

items = []

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

9 return '75:29:55:73'

Вариант: 2-2-30

1. gcd(-9, -75) = 3

1 def gcd(x=-9, y=-75)

2 if -9 < 0: --- True

3 x = --9

x = 9

4 if -75 < 0: --- True

5 y = --75

y = 75

6 if 9 == 0: --- False

8 while 75 != 0: --- True

9 rem = 9 % 75

rem = 9

10 x = 75

11 y = 9

8 while 9 != 0: --- True

9 rem = 75 % 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(-65, 0) = 65

1 def gcd(x=-65, y=0)

2 if -65 < 0: --- True

3 x = --65

x = 65

4 if 0 < 0: --- False

6 if 65 == 0: --- False

8 while 0 != 0: --- False

12 return 65

3. hex(183) = 'B7'

3 def hex(number=183)

4 if 183 == 0: --- False

6 res = ''

7 while 183 > 0: --- True

8 digit = 183 % 16

digit = 7

9 res = DIGITS[7] + ''

res = '7'

10 number = 183 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + '7'

res = 'B7'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'B7'

4. square\_equal(0, -100, -36) = [-0.36]

3 def square\_equal(a=0, b=-100, c=-36)

4 if 0 != 0: --- False

14 else:

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

16 return [-0.36]

5. square\_equal(-83, 15, -38) = []

3 def square\_equal(a=-83, b=15, c=-38)

4 if -83 != 0: --- True

5 D = 15\*15 - 4\*-83\*-38

D = -12391

6 if -12391 > 0: --- False

10 elif -12391 == 0: --- False

12 else:

13 return []

6. findmax([-28, -26, 35, -86]) = 35

1 def findmax(items=[-28, -26, 35, -86])

2 if len([-28, -26, 35, -86]) == 0: --- False

4 m = items[0]

m = -28

5 i = 1

6 while 1 < len([-28, -26, 35, -86]): --- True

7 if -28 < items[1]: --- True

8 m = items[1]

m = -26

9 i = 1 + 1

i = 2

6 while 2 < len([-28, -26, 35, -86]): --- True

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

8 m = items[2]

m = 35

9 i = 2 + 1

i = 3

6 while 3 < len([-28, -26, 35, -86]): --- True

7 if 35 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-28, -26, 35, -86]): --- False

10 return 35

7. unique([14, 28, 28, 79]) = [14, 28, 79]

1 def unique(items=[14, 28, 28, 79])

2 res = []

3 i = 0

4 while 0 < len([14, 28, 28, 79]): --- 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, 28, 28, 79]): --- True

5 if len([14]) == 0 or res[-1] != items[1]: --- True

6 res = [14] + [items[1]]

res = [14, 28]

7 i = 1 + 1

i = 2

4 while 2 < len([14, 28, 28, 79]): --- True

5 if len([14, 28]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([14, 28, 28, 79]): --- True

5 if len([14, 28]) == 0 or res[-1] != items[3]: --- True

6 res = [14, 28] + [items[3]]

res = [14, 28, 79]

7 i = 3 + 1

i = 4

4 while 4 < len([14, 28, 28, 79]): --- False

8 return [14, 28, 79]

8. join('+', [46, 11, 56]) = '46+11+56'

1 def join(sep=+, items=[46, 11, 56])

2 res = ''

3 if len([46, 11, 56]) > 0: --- True

4 res = str(items[0])

res = '46'

5 items = items[1:]

items = [11, 56]

6 while len([11, 56]) > 0: --- True

7 res = '46' + '+' + str(items[0])

res = '46+11'

8 items = items[1:]

items = [56]

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

7 res = '46+11' + '+' + str(items[0])

res = '46+11+56'

8 items = items[1:]

items = []

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

9 return '46+11+56'