Вариант: 2-4-1

1. gcd(32, -48) = 16

1 def gcd(x=32, y=-48)

2 if 32 < 0: --- False

4 if -48 < 0: --- True

5 y = --48

y = 48

6 if 32 == 0: --- False

8 while 48 != 0: --- True

9 rem = 32 % 48

rem = 32

10 x = 48

11 y = 32

8 while 32 != 0: --- True

9 rem = 48 % 32

rem = 16

10 x = 32

11 y = 16

8 while 16 != 0: --- True

9 rem = 32 % 16

rem = 0

10 x = 16

11 y = 0

8 while 0 != 0: --- False

12 return 16

2. gcd(0, -28) = 28

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

2 if 0 < 0: --- False

4 if -28 < 0: --- True

5 y = --28

y = 28

6 if 0 == 0: --- True

7 return 28

3. hex(255) = 'FF'

3 def hex(number=255)

4 if 255 == 0: --- False

6 res = ''

7 while 255 > 0: --- True

8 digit = 255 % 16

digit = 15

9 res = DIGITS[15] + ''

res = 'F'

10 number = 255 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + 'F'

res = 'FF'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'FF'

4. square\_equal(10, 8, 0) = [-0.8, 0.0]

3 def square\_equal(a=10, b=8, c=0)

4 if 10 != 0: --- True

5 D = 8\*8 - 4\*10\*0

D = 64

6 if 64 > 0: --- True

7 x1 = (-8 - sqrt(64)) / (2\*10)

x1 = -0.8

8 x2 = (-8 + sqrt(64)) / (2\*10)

x2 = 0.0

9 return [-0.8, 0.0]

5. square\_equal(-85, 19, -85) = []

3 def square\_equal(a=-85, b=19, c=-85)

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

5 D = 19\*19 - 4\*-85\*-85

D = -28539

6 if -28539 > 0: --- False

10 elif -28539 == 0: --- False

12 else:

13 return []

6. findmax([92, -20, -76, 44, 96]) = 96

1 def findmax(items=[92, -20, -76, 44, 96])

2 if len([92, -20, -76, 44, 96]) == 0: --- False

4 m = items[0]

m = 92

5 i = 1

6 while 1 < len([92, -20, -76, 44, 96]): --- True

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

9 i = 1 + 1

i = 2

6 while 2 < len([92, -20, -76, 44, 96]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([92, -20, -76, 44, 96]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([92, -20, -76, 44, 96]): --- True

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

8 m = items[4]

m = 96

9 i = 4 + 1

i = 5

6 while 5 < len([92, -20, -76, 44, 96]): --- False

10 return 96

7. unique([55, 55, -41]) = [55, -41]

1 def unique(items=[55, 55, -41])

2 res = []

3 i = 0

4 while 0 < len([55, 55, -41]): --- True

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

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

res = [55]

7 i = 0 + 1

i = 1

4 while 1 < len([55, 55, -41]): --- True

5 if len([55]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([55, 55, -41]): --- True

5 if len([55]) == 0 or res[-1] != items[2]: --- True

6 res = [55] + [items[2]]

res = [55, -41]

7 i = 2 + 1

i = 3

4 while 3 < len([55, 55, -41]): --- False

8 return [55, -41]

8. join(';', [24, 91, 32]) = '24;91;32'

1 def join(sep=';', items=[24, 91, 32])

2 res = ''

3 if len([24, 91, 32]) > 0: --- True

4 res = str(items[0])

res = '24'

5 items = items[1:]

items = [91, 32]

6 while len([91, 32]) > 0: --- True

7 res = '24' + ';' + str(items[0])

res = '24;91'

8 items = items[1:]

items = [32]

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

7 res = '24;91' + ';' + str(items[0])

res = '24;91;32'

8 items = items[1:]

items = []

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

9 return '24;91;32'

Вариант: 2-4-2

1. gcd(60, 36) = 12

1 def gcd(x=60, y=36)

2 if 60 < 0: --- False

4 if 36 < 0: --- False

6 if 60 == 0: --- False

8 while 36 != 0: --- True

9 rem = 60 % 36

rem = 24

10 x = 36

11 y = 24

8 while 24 != 0: --- True

9 rem = 36 % 24

rem = 12

10 x = 24

11 y = 12

8 while 12 != 0: --- True

9 rem = 24 % 12

rem = 0

10 x = 12

11 y = 0

8 while 0 != 0: --- False

12 return 12

2. gcd(0, -25) = 25

1 def gcd(x=0, y=-25)

2 if 0 < 0: --- False

4 if -25 < 0: --- True

5 y = --25

y = 25

6 if 0 == 0: --- True

7 return 25

3. hex(232) = 'E8'

3 def hex(number=232)

4 if 232 == 0: --- False

6 res = ''

7 while 232 > 0: --- True

8 digit = 232 % 16

digit = 8

9 res = DIGITS[8] + ''

res = '8'

10 number = 232 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + '8'

res = 'E8'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'E8'

4. square\_equal(20, 34, -12) = [-2.0, 0.3]

3 def square\_equal(a=20, b=34, c=-12)

4 if 20 != 0: --- True

5 D = 34\*34 - 4\*20\*-12

D = 2116

6 if 2116 > 0: --- True

7 x1 = (-34 - sqrt(2116)) / (2\*20)

x1 = -2.0

8 x2 = (-34 + sqrt(2116)) / (2\*20)

x2 = 0.3

9 return [-2.0, 0.3]

5. square\_equal(61, -69, 85) = []

3 def square\_equal(a=61, b=-69, c=85)

4 if 61 != 0: --- True

5 D = -69\*-69 - 4\*61\*85

D = -15979

6 if -15979 > 0: --- False

10 elif -15979 == 0: --- False

12 else:

13 return []

6. findmax([-54, 99, -28, 60]) = 99

1 def findmax(items=[-54, 99, -28, 60])

2 if len([-54, 99, -28, 60]) == 0: --- False

4 m = items[0]

m = -54

5 i = 1

6 while 1 < len([-54, 99, -28, 60]): --- True

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

8 m = items[1]

m = 99

9 i = 1 + 1

i = 2

6 while 2 < len([-54, 99, -28, 60]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([-54, 99, -28, 60]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([-54, 99, -28, 60]): --- False

10 return 99

7. unique([-53, 23, -90, -90]) = [-53, 23, -90]

1 def unique(items=[-53, 23, -90, -90])

2 res = []

3 i = 0

4 while 0 < len([-53, 23, -90, -90]): --- True

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

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

res = [-53]

7 i = 0 + 1

i = 1

4 while 1 < len([-53, 23, -90, -90]): --- True

5 if len([-53]) == 0 or res[-1] != items[1]: --- True

6 res = [-53] + [items[1]]

res = [-53, 23]

7 i = 1 + 1

i = 2

4 while 2 < len([-53, 23, -90, -90]): --- True

5 if len([-53, 23]) == 0 or res[-1] != items[2]: --- True

6 res = [-53, 23] + [items[2]]

res = [-53, 23, -90]

7 i = 2 + 1

i = 3

4 while 3 < len([-53, 23, -90, -90]): --- True

5 if len([-53, 23, -90]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([-53, 23, -90, -90]): --- False

8 return [-53, 23, -90]

8. join(':', [19, 37, 50]) = '19:37:50'

1 def join(sep=':', items=[19, 37, 50])

2 res = ''

3 if len([19, 37, 50]) > 0: --- True

4 res = str(items[0])

res = '19'

5 items = items[1:]

items = [37, 50]

6 while len([37, 50]) > 0: --- True

7 res = '19' + ':' + str(items[0])

res = '19:37'

8 items = items[1:]

items = [50]

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

7 res = '19:37' + ':' + str(items[0])

res = '19:37:50'

8 items = items[1:]

items = []

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

9 return '19:37:50'

Вариант: 2-4-3

1. gcd(51, 15) = 3

1 def gcd(x=51, y=15)

2 if 51 < 0: --- False

4 if 15 < 0: --- False

6 if 51 == 0: --- False

8 while 15 != 0: --- True

9 rem = 51 % 15

rem = 6

10 x = 15

11 y = 6

8 while 6 != 0: --- True

9 rem = 15 % 6

rem = 3

10 x = 6

11 y = 3

8 while 3 != 0: --- True

9 rem = 6 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(0, -2) = 2

1 def gcd(x=0, y=-2)

2 if 0 < 0: --- False

4 if -2 < 0: --- True

5 y = --2

y = 2

6 if 0 == 0: --- True

7 return 2

3. hex(244) = 'F4'

3 def hex(number=244)

4 if 244 == 0: --- False

6 res = ''

7 while 244 > 0: --- True

8 digit = 244 % 16

digit = 4

9 res = DIGITS[4] + ''

res = '4'

10 number = 244 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + '4'

res = 'F4'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'F4'

4. square\_equal(-50, 25, 3) = [0.6, -0.1]

3 def square\_equal(a=-50, b=25, c=3)

4 if -50 != 0: --- True

5 D = 25\*25 - 4\*-50\*3

D = 1225

6 if 1225 > 0: --- True

7 x1 = (-25 - sqrt(1225)) / (2\*-50)

x1 = 0.6

8 x2 = (-25 + sqrt(1225)) / (2\*-50)

x2 = -0.1

9 return [0.6, -0.1]

5. square\_equal(-36, -20, -35) = []

3 def square\_equal(a=-36, b=-20, c=-35)

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

5 D = -20\*-20 - 4\*-36\*-35

D = -4640

6 if -4640 > 0: --- False

10 elif -4640 == 0: --- False

12 else:

13 return []

6. findmax([-79, 74, -66, 63]) = 74

1 def findmax(items=[-79, 74, -66, 63])

2 if len([-79, 74, -66, 63]) == 0: --- False

4 m = items[0]

m = -79

5 i = 1

6 while 1 < len([-79, 74, -66, 63]): --- True

7 if -79 < items[1]: --- True

8 m = items[1]

m = 74

9 i = 1 + 1

i = 2

6 while 2 < len([-79, 74, -66, 63]): --- True

7 if 74 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-79, 74, -66, 63]): --- True

7 if 74 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([-79, 74, -66, 63]): --- False

10 return 74

7. unique([-34, -34, 9, -34]) = [-34, 9, -34]

1 def unique(items=[-34, -34, 9, -34])

2 res = []

3 i = 0

4 while 0 < len([-34, -34, 9, -34]): --- True

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

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

res = [-34]

7 i = 0 + 1

i = 1

4 while 1 < len([-34, -34, 9, -34]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([-34, -34, 9, -34]): --- True

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

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

res = [-34, 9]

7 i = 2 + 1

i = 3

4 while 3 < len([-34, -34, 9, -34]): --- True

5 if len([-34, 9]) == 0 or res[-1] != items[3]: --- True

6 res = [-34, 9] + [items[3]]

res = [-34, 9, -34]

7 i = 3 + 1

i = 4

4 while 4 < len([-34, -34, 9, -34]): --- False

8 return [-34, 9, -34]

8. join(';', [98, 83, 49, 6]) = '98;83;49;6'

1 def join(sep=';', items=[98, 83, 49, 6])

2 res = ''

3 if len([98, 83, 49, 6]) > 0: --- True

4 res = str(items[0])

res = '98'

5 items = items[1:]

items = [83, 49, 6]

6 while len([83, 49, 6]) > 0: --- True

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

res = '98;83'

8 items = items[1:]

items = [49, 6]

6 while len([49, 6]) > 0: --- True

7 res = '98;83' + ';' + str(items[0])

res = '98;83;49'

8 items = items[1:]

items = [6]

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

7 res = '98;83;49' + ';' + str(items[0])

res = '98;83;49;6'

8 items = items[1:]

items = []

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

9 return '98;83;49;6'

Вариант: 2-4-4

1. gcd(81, -75) = 3

1 def gcd(x=81, y=-75)

2 if 81 < 0: --- False

4 if -75 < 0: --- True

5 y = --75

y = 75

6 if 81 == 0: --- False

8 while 75 != 0: --- True

9 rem = 81 % 75

rem = 6

10 x = 75

11 y = 6

8 while 6 != 0: --- True

9 rem = 75 % 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(91, 0) = 91

1 def gcd(x=91, y=0)

2 if 91 < 0: --- False

4 if 0 < 0: --- False

6 if 91 == 0: --- False

8 while 0 != 0: --- False

12 return 91

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(-25, 34, 59) = [2.36, -1.0]

3 def square\_equal(a=-25, b=34, c=59)

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

5 D = 34\*34 - 4\*-25\*59

D = 7056

6 if 7056 > 0: --- True

7 x1 = (-34 - sqrt(7056)) / (2\*-25)

x1 = 2.36

8 x2 = (-34 + sqrt(7056)) / (2\*-25)

x2 = -1.0

9 return [2.36, -1.0]

5. square\_equal(97, 91, 97) = []

3 def square\_equal(a=97, b=91, c=97)

4 if 97 != 0: --- True

5 D = 91\*91 - 4\*97\*97

D = -29355

6 if -29355 > 0: --- False

10 elif -29355 == 0: --- False

12 else:

13 return []

6. findmax([83, -81, -35, 43, 55, -5]) = 83

1 def findmax(items=[83, -81, -35, 43, 55, -5])

2 if len([83, -81, -35, 43, 55, -5]) == 0: --- False

4 m = items[0]

m = 83

5 i = 1

6 while 1 < len([83, -81, -35, 43, 55, -5]): --- True

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

9 i = 1 + 1

i = 2

6 while 2 < len([83, -81, -35, 43, 55, -5]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([83, -81, -35, 43, 55, -5]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([83, -81, -35, 43, 55, -5]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([83, -81, -35, 43, 55, -5]): --- True

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

9 i = 5 + 1

i = 6

6 while 6 < len([83, -81, -35, 43, 55, -5]): --- False

10 return 83

7. unique([-9, -29, -29]) = [-9, -29]

1 def unique(items=[-9, -29, -29])

2 res = []

3 i = 0

4 while 0 < len([-9, -29, -29]): --- True

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

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

res = [-9]

7 i = 0 + 1

i = 1

4 while 1 < len([-9, -29, -29]): --- True

5 if len([-9]) == 0 or res[-1] != items[1]: --- True

6 res = [-9] + [items[1]]

res = [-9, -29]

7 i = 1 + 1

i = 2

4 while 2 < len([-9, -29, -29]): --- True

5 if len([-9, -29]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-9, -29, -29]): --- False

8 return [-9, -29]

8. join('+', [24, 10, 98]) = '24+10+98'

1 def join(sep='+', items=[24, 10, 98])

2 res = ''

3 if len([24, 10, 98]) > 0: --- True

4 res = str(items[0])

res = '24'

5 items = items[1:]

items = [10, 98]

6 while len([10, 98]) > 0: --- True

7 res = '24' + '+' + str(items[0])

res = '24+10'

8 items = items[1:]

items = [98]

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

7 res = '24+10' + '+' + str(items[0])

res = '24+10+98'

8 items = items[1:]

items = []

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

9 return '24+10+98'

Вариант: 2-4-5

1. gcd(-80, 12) = 4

1 def gcd(x=-80, y=12)

2 if -80 < 0: --- True

3 x = --80

x = 80

4 if 12 < 0: --- False

6 if 80 == 0: --- False

8 while 12 != 0: --- True

9 rem = 80 % 12

rem = 8

10 x = 12

11 y = 8

8 while 8 != 0: --- True

9 rem = 12 % 8

rem = 4

10 x = 8

11 y = 4

8 while 4 != 0: --- True

9 rem = 8 % 4

rem = 0

10 x = 4

11 y = 0

8 while 0 != 0: --- False

12 return 4

2. gcd(2, 0) = 2

1 def gcd(x=2, y=0)

2 if 2 < 0: --- False

4 if 0 < 0: --- False

6 if 2 == 0: --- False

8 while 0 != 0: --- False

12 return 2

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(-20, -37, 69) = [1.15, -3.0]

3 def square\_equal(a=-20, b=-37, c=69)

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

5 D = -37\*-37 - 4\*-20\*69

D = 6889

6 if 6889 > 0: --- True

7 x1 = (--37 - sqrt(6889)) / (2\*-20)

x1 = 1.15

8 x2 = (--37 + sqrt(6889)) / (2\*-20)

x2 = -3.0

9 return [1.15, -3.0]

5. square\_equal(28, 86, 70) = []

3 def square\_equal(a=28, b=86, c=70)

4 if 28 != 0: --- True

5 D = 86\*86 - 4\*28\*70

D = -444

6 if -444 > 0: --- False

10 elif -444 == 0: --- False

12 else:

13 return []

6. findmax([34, 67, -9, -85, 93]) = 93

1 def findmax(items=[34, 67, -9, -85, 93])

2 if len([34, 67, -9, -85, 93]) == 0: --- False

4 m = items[0]

m = 34

5 i = 1

6 while 1 < len([34, 67, -9, -85, 93]): --- True

7 if 34 < items[1]: --- True

8 m = items[1]

m = 67

9 i = 1 + 1

i = 2

6 while 2 < len([34, 67, -9, -85, 93]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([34, 67, -9, -85, 93]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([34, 67, -9, -85, 93]): --- True

7 if 67 < items[4]: --- True

8 m = items[4]

m = 93

9 i = 4 + 1

i = 5

6 while 5 < len([34, 67, -9, -85, 93]): --- False

10 return 93

7. unique([-67, -67, 57, -67]) = [-67, 57, -67]

1 def unique(items=[-67, -67, 57, -67])

2 res = []

3 i = 0

4 while 0 < len([-67, -67, 57, -67]): --- True

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

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

res = [-67]

7 i = 0 + 1

i = 1

4 while 1 < len([-67, -67, 57, -67]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([-67, -67, 57, -67]): --- True

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

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

res = [-67, 57]

7 i = 2 + 1

i = 3

4 while 3 < len([-67, -67, 57, -67]): --- True

5 if len([-67, 57]) == 0 or res[-1] != items[3]: --- True

6 res = [-67, 57] + [items[3]]

res = [-67, 57, -67]

7 i = 3 + 1

i = 4

4 while 4 < len([-67, -67, 57, -67]): --- False

8 return [-67, 57, -67]

8. join(':', [25, 55, 58]) = '25:55:58'

1 def join(sep=':', items=[25, 55, 58])

2 res = ''

3 if len([25, 55, 58]) > 0: --- True

4 res = str(items[0])

res = '25'

5 items = items[1:]

items = [55, 58]

6 while len([55, 58]) > 0: --- True

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

res = '25:55'

8 items = items[1:]

items = [58]

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

7 res = '25:55' + ':' + str(items[0])

res = '25:55:58'

8 items = items[1:]

items = []

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

9 return '25:55:58'

Вариант: 2-4-6

1. gcd(-99, -88) = 11

1 def gcd(x=-99, y=-88)

2 if -99 < 0: --- True

3 x = --99

x = 99

4 if -88 < 0: --- True

5 y = --88

y = 88

6 if 99 == 0: --- False

8 while 88 != 0: --- True

9 rem = 99 % 88

rem = 11

10 x = 88

11 y = 11

8 while 11 != 0: --- True

9 rem = 88 % 11

rem = 0

10 x = 11

11 y = 0

8 while 0 != 0: --- False

12 return 11

2. gcd(54, 0) = 54

1 def gcd(x=54, y=0)

2 if 54 < 0: --- False

4 if 0 < 0: --- False

6 if 54 == 0: --- False

8 while 0 != 0: --- False

12 return 54

3. hex(248) = 'F8'

3 def hex(number=248)

4 if 248 == 0: --- False

6 res = ''

7 while 248 > 0: --- True

8 digit = 248 % 16

digit = 8

9 res = DIGITS[8] + ''

res = '8'

10 number = 248 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + '8'

res = 'F8'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'F8'

4. square\_equal(4, 18, 18) = [-3.0, -1.5]

3 def square\_equal(a=4, b=18, c=18)

4 if 4 != 0: --- True

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

D = 36

6 if 36 > 0: --- True

7 x1 = (-18 - sqrt(36)) / (2\*4)

x1 = -3.0

8 x2 = (-18 + sqrt(36)) / (2\*4)

x2 = -1.5

9 return [-3.0, -1.5]

5. square\_equal(72, 50, 26) = []

3 def square\_equal(a=72, b=50, c=26)

4 if 72 != 0: --- True

5 D = 50\*50 - 4\*72\*26

D = -4988

6 if -4988 > 0: --- False

10 elif -4988 == 0: --- False

12 else:

13 return []

6. findmax([99, -85, -73, 11, -10]) = 99

1 def findmax(items=[99, -85, -73, 11, -10])

2 if len([99, -85, -73, 11, -10]) == 0: --- False

4 m = items[0]

m = 99

5 i = 1

6 while 1 < len([99, -85, -73, 11, -10]): --- True

7 if 99 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([99, -85, -73, 11, -10]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([99, -85, -73, 11, -10]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([99, -85, -73, 11, -10]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([99, -85, -73, 11, -10]): --- False

10 return 99

7. unique([-57, -57, 84, -57]) = [-57, 84, -57]

1 def unique(items=[-57, -57, 84, -57])

2 res = []

3 i = 0

4 while 0 < len([-57, -57, 84, -57]): --- True

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

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

res = [-57]

7 i = 0 + 1

i = 1

4 while 1 < len([-57, -57, 84, -57]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([-57, -57, 84, -57]): --- True

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

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

res = [-57, 84]

7 i = 2 + 1

i = 3

4 while 3 < len([-57, -57, 84, -57]): --- True

5 if len([-57, 84]) == 0 or res[-1] != items[3]: --- True

6 res = [-57, 84] + [items[3]]

res = [-57, 84, -57]

7 i = 3 + 1

i = 4

4 while 4 < len([-57, -57, 84, -57]): --- False

8 return [-57, 84, -57]

8. join(',', [87, 75, 30, 2]) = '87,75,30,2'

1 def join(sep=',', items=[87, 75, 30, 2])

2 res = ''

3 if len([87, 75, 30, 2]) > 0: --- True

4 res = str(items[0])

res = '87'

5 items = items[1:]

items = [75, 30, 2]

6 while len([75, 30, 2]) > 0: --- True

7 res = '87' + ',' + str(items[0])

res = '87,75'

8 items = items[1:]

items = [30, 2]

6 while len([30, 2]) > 0: --- True

7 res = '87,75' + ',' + str(items[0])

res = '87,75,30'

8 items = items[1:]

items = [2]

6 while len([2]) > 0: --- True

7 res = '87,75,30' + ',' + str(items[0])

res = '87,75,30,2'

8 items = items[1:]

items = []

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

9 return '87,75,30,2'

Вариант: 2-4-7

1. gcd(-75, 93) = 3

1 def gcd(x=-75, y=93)

2 if -75 < 0: --- True

3 x = --75

x = 75

4 if 93 < 0: --- False

6 if 75 == 0: --- False

8 while 93 != 0: --- True

9 rem = 75 % 93

rem = 75

10 x = 93

11 y = 75

8 while 75 != 0: --- True

9 rem = 93 % 75

rem = 18

10 x = 75

11 y = 18

8 while 18 != 0: --- True

9 rem = 75 % 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, -99) = 99

1 def gcd(x=0, y=-99)

2 if 0 < 0: --- False

4 if -99 < 0: --- True

5 y = --99

y = 99

6 if 0 == 0: --- True

7 return 99

3. hex(167) = 'A7'

3 def hex(number=167)

4 if 167 == 0: --- False

6 res = ''

7 while 167 > 0: --- True

8 digit = 167 % 16

digit = 7

9 res = DIGITS[7] + ''

res = '7'

10 number = 167 // 16

number = 10

7 while 10 > 0: --- True

8 digit = 10 % 16

digit = 10

9 res = DIGITS[10] + '7'

res = 'A7'

10 number = 10 // 16

number = 0

7 while 0 > 0: --- False

11 return 'A7'

4. square\_equal(25, 27, -46) = [-2.0, 0.92]

3 def square\_equal(a=25, b=27, c=-46)

4 if 25 != 0: --- True

5 D = 27\*27 - 4\*25\*-46

D = 5329

6 if 5329 > 0: --- True

7 x1 = (-27 - sqrt(5329)) / (2\*25)

x1 = -2.0

8 x2 = (-27 + sqrt(5329)) / (2\*25)

x2 = 0.92

9 return [-2.0, 0.92]

5. square\_equal(34, 53, 84) = []

3 def square\_equal(a=34, b=53, c=84)

4 if 34 != 0: --- True

5 D = 53\*53 - 4\*34\*84

D = -8615

6 if -8615 > 0: --- False

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

12 else:

13 return []

6. findmax([-35, 28, -51, 62]) = 62

1 def findmax(items=[-35, 28, -51, 62])

2 if len([-35, 28, -51, 62]) == 0: --- False

4 m = items[0]

m = -35

5 i = 1

6 while 1 < len([-35, 28, -51, 62]): --- True

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

8 m = items[1]

m = 28

9 i = 1 + 1

i = 2

6 while 2 < len([-35, 28, -51, 62]): --- True

7 if 28 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-35, 28, -51, 62]): --- True

7 if 28 < items[3]: --- True

8 m = items[3]

m = 62

9 i = 3 + 1

i = 4

6 while 4 < len([-35, 28, -51, 62]): --- False

10 return 62

7. unique([-96, -30, -96, -96]) = [-96, -30, -96]

1 def unique(items=[-96, -30, -96, -96])

2 res = []

3 i = 0

4 while 0 < len([-96, -30, -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, -30, -96, -96]): --- True

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

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

res = [-96, -30]

7 i = 1 + 1

i = 2

4 while 2 < len([-96, -30, -96, -96]): --- True

5 if len([-96, -30]) == 0 or res[-1] != items[2]: --- True

6 res = [-96, -30] + [items[2]]

res = [-96, -30, -96]

7 i = 2 + 1

i = 3

4 while 3 < len([-96, -30, -96, -96]): --- True

5 if len([-96, -30, -96]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([-96, -30, -96, -96]): --- False

8 return [-96, -30, -96]

8. join('+', [2, 2, 44, 88]) = '2+2+44+88'

1 def join(sep='+', items=[2, 2, 44, 88])

2 res = ''

3 if len([2, 2, 44, 88]) > 0: --- True

4 res = str(items[0])

res = '2'

5 items = items[1:]

items = [2, 44, 88]

6 while len([2, 44, 88]) > 0: --- True

7 res = '2' + '+' + str(items[0])

res = '2+2'

8 items = items[1:]

items = [44, 88]

6 while len([44, 88]) > 0: --- True

7 res = '2+2' + '+' + str(items[0])

res = '2+2+44'

8 items = items[1:]

items = [88]

6 while len([88]) > 0: --- True

7 res = '2+2+44' + '+' + str(items[0])

res = '2+2+44+88'

8 items = items[1:]

items = []

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

9 return '2+2+44+88'

Вариант: 2-4-8

1. gcd(80, 96) = 16

1 def gcd(x=80, y=96)

2 if 80 < 0: --- False

4 if 96 < 0: --- False

6 if 80 == 0: --- False

8 while 96 != 0: --- True

9 rem = 80 % 96

rem = 80

10 x = 96

11 y = 80

8 while 80 != 0: --- True

9 rem = 96 % 80

rem = 16

10 x = 80

11 y = 16

8 while 16 != 0: --- True

9 rem = 80 % 16

rem = 0

10 x = 16

11 y = 0

8 while 0 != 0: --- False

12 return 16

2. gcd(80, 0) = 80

1 def gcd(x=80, y=0)

2 if 80 < 0: --- False

4 if 0 < 0: --- False

6 if 80 == 0: --- False

8 while 0 != 0: --- False

12 return 80

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(-4, -40, -64) = [-2.0, -8.0]

3 def square\_equal(a=-4, b=-40, c=-64)

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

5 D = -40\*-40 - 4\*-4\*-64

D = 576

6 if 576 > 0: --- True

7 x1 = (--40 - sqrt(576)) / (2\*-4)

x1 = -2.0

8 x2 = (--40 + sqrt(576)) / (2\*-4)

x2 = -8.0

9 return [-2.0, -8.0]

5. square\_equal(-57, -23, -50) = []

3 def square\_equal(a=-57, b=-23, c=-50)

4 if -57 != 0: --- True

5 D = -23\*-23 - 4\*-57\*-50

D = -10871

6 if -10871 > 0: --- False

10 elif -10871 == 0: --- False

12 else:

13 return []

6. findmax([-48, 13, -57, 58, -42]) = 58

1 def findmax(items=[-48, 13, -57, 58, -42])

2 if len([-48, 13, -57, 58, -42]) == 0: --- False

4 m = items[0]

m = -48

5 i = 1

6 while 1 < len([-48, 13, -57, 58, -42]): --- True

7 if -48 < items[1]: --- True

8 m = items[1]

m = 13

9 i = 1 + 1

i = 2

6 while 2 < len([-48, 13, -57, 58, -42]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([-48, 13, -57, 58, -42]): --- True

7 if 13 < items[3]: --- True

8 m = items[3]

m = 58

9 i = 3 + 1

i = 4

6 while 4 < len([-48, 13, -57, 58, -42]): --- True

7 if 58 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([-48, 13, -57, 58, -42]): --- False

10 return 58

7. unique([-82, -82, 76, -16]) = [-82, 76, -16]

1 def unique(items=[-82, -82, 76, -16])

2 res = []

3 i = 0

4 while 0 < len([-82, -82, 76, -16]): --- True

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

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

res = [-82]

7 i = 0 + 1

i = 1

4 while 1 < len([-82, -82, 76, -16]): --- True

5 if len([-82]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([-82, -82, 76, -16]): --- True

5 if len([-82]) == 0 or res[-1] != items[2]: --- True

6 res = [-82] + [items[2]]

res = [-82, 76]

7 i = 2 + 1

i = 3

4 while 3 < len([-82, -82, 76, -16]): --- True

5 if len([-82, 76]) == 0 or res[-1] != items[3]: --- True

6 res = [-82, 76] + [items[3]]

res = [-82, 76, -16]

7 i = 3 + 1

i = 4

4 while 4 < len([-82, -82, 76, -16]): --- False

8 return [-82, 76, -16]

8. join(':', [94, 3, 23, 62]) = '94:3:23:62'

1 def join(sep=':', items=[94, 3, 23, 62])

2 res = ''

3 if len([94, 3, 23, 62]) > 0: --- True

4 res = str(items[0])

res = '94'

5 items = items[1:]

items = [3, 23, 62]

6 while len([3, 23, 62]) > 0: --- True

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

res = '94:3'

8 items = items[1:]

items = [23, 62]

6 while len([23, 62]) > 0: --- True

7 res = '94:3' + ':' + str(items[0])

res = '94:3:23'

8 items = items[1:]

items = [62]

6 while len([62]) > 0: --- True

7 res = '94:3:23' + ':' + str(items[0])

res = '94:3:23:62'

8 items = items[1:]

items = []

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

9 return '94:3:23:62'

Вариант: 2-4-9

1. gcd(9, -69) = 3

1 def gcd(x=9, y=-69)

2 if 9 < 0: --- False

4 if -69 < 0: --- True

5 y = --69

y = 69

6 if 9 == 0: --- False

8 while 69 != 0: --- True

9 rem = 9 % 69

rem = 9

10 x = 69

11 y = 9

8 while 9 != 0: --- True

9 rem = 69 % 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(-82, 0) = 82

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

2 if -82 < 0: --- True

3 x = --82

x = 82

4 if 0 < 0: --- False

6 if 82 == 0: --- False

8 while 0 != 0: --- False

12 return 82

3. hex(213) = 'D5'

3 def hex(number=213)

4 if 213 == 0: --- False

6 res = ''

7 while 213 > 0: --- True

8 digit = 213 % 16

digit = 5

9 res = DIGITS[5] + ''

res = '5'

10 number = 213 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + '5'

res = 'D5'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'D5'

4. square\_equal(100, -95, 0) = [0.0, 0.95]

3 def square\_equal(a=100, b=-95, c=0)

4 if 100 != 0: --- True

5 D = -95\*-95 - 4\*100\*0

D = 9025

6 if 9025 > 0: --- True

7 x1 = (--95 - sqrt(9025)) / (2\*100)

x1 = 0.0

8 x2 = (--95 + sqrt(9025)) / (2\*100)

x2 = 0.95

9 return [0.0, 0.95]

5. square\_equal(-43, -70, -75) = []

3 def square\_equal(a=-43, b=-70, c=-75)

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

5 D = -70\*-70 - 4\*-43\*-75

D = -8000

6 if -8000 > 0: --- False

10 elif -8000 == 0: --- False

12 else:

13 return []

6. findmax([7, 67, 15, -59, -88, -23]) = 67

1 def findmax(items=[7, 67, 15, -59, -88, -23])

2 if len([7, 67, 15, -59, -88, -23]) == 0: --- False

4 m = items[0]

m = 7

5 i = 1

6 while 1 < len([7, 67, 15, -59, -88, -23]): --- True

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

8 m = items[1]

m = 67

9 i = 1 + 1

i = 2

6 while 2 < len([7, 67, 15, -59, -88, -23]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([7, 67, 15, -59, -88, -23]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([7, 67, 15, -59, -88, -23]): --- True

7 if 67 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([7, 67, 15, -59, -88, -23]): --- True

7 if 67 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([7, 67, 15, -59, -88, -23]): --- False

10 return 67

7. unique([79, 79, 12, -74]) = [79, 12, -74]

1 def unique(items=[79, 79, 12, -74])

2 res = []

3 i = 0

4 while 0 < len([79, 79, 12, -74]): --- True

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

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

res = [79]

7 i = 0 + 1

i = 1

4 while 1 < len([79, 79, 12, -74]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([79, 79, 12, -74]): --- True

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

6 res = [79] + [items[2]]

res = [79, 12]

7 i = 2 + 1

i = 3

4 while 3 < len([79, 79, 12, -74]): --- True

5 if len([79, 12]) == 0 or res[-1] != items[3]: --- True

6 res = [79, 12] + [items[3]]

res = [79, 12, -74]

7 i = 3 + 1

i = 4

4 while 4 < len([79, 79, 12, -74]): --- False

8 return [79, 12, -74]

8. join(';', [42, 58, 74]) = '42;58;74'

1 def join(sep=';', items=[42, 58, 74])

2 res = ''

3 if len([42, 58, 74]) > 0: --- True

4 res = str(items[0])

res = '42'

5 items = items[1:]

items = [58, 74]

6 while len([58, 74]) > 0: --- True

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

res = '42;58'

8 items = items[1:]

items = [74]

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

7 res = '42;58' + ';' + str(items[0])

res = '42;58;74'

8 items = items[1:]

items = []

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

9 return '42;58;74'

Вариант: 2-4-10

1. gcd(-40, 70) = 10

1 def gcd(x=-40, y=70)

2 if -40 < 0: --- True

3 x = --40

x = 40

4 if 70 < 0: --- False

6 if 40 == 0: --- False

8 while 70 != 0: --- True

9 rem = 40 % 70

rem = 40

10 x = 70

11 y = 40

8 while 40 != 0: --- True

9 rem = 70 % 40

rem = 30

10 x = 40

11 y = 30

8 while 30 != 0: --- True

9 rem = 40 % 30

rem = 10

10 x = 30

11 y = 10

8 while 10 != 0: --- True

9 rem = 30 % 10

rem = 0

10 x = 10

11 y = 0

8 while 0 != 0: --- False

12 return 10

2. gcd(0, 87) = 87

1 def gcd(x=0, y=87)

2 if 0 < 0: --- False

4 if 87 < 0: --- False

6 if 0 == 0: --- True

7 return 87

3. hex(178) = 'B2'

3 def hex(number=178)

4 if 178 == 0: --- False

6 res = ''

7 while 178 > 0: --- True

8 digit = 178 % 16

digit = 2

9 res = DIGITS[2] + ''

res = '2'

10 number = 178 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + '2'

res = 'B2'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'B2'

4. square\_equal(25, -40, -65) = [-1.0, 2.6]

3 def square\_equal(a=25, b=-40, c=-65)

4 if 25 != 0: --- True

5 D = -40\*-40 - 4\*25\*-65

D = 8100

6 if 8100 > 0: --- True

7 x1 = (--40 - sqrt(8100)) / (2\*25)

x1 = -1.0

8 x2 = (--40 + sqrt(8100)) / (2\*25)

x2 = 2.6

9 return [-1.0, 2.6]

5. square\_equal(-49, -99, -56) = []

3 def square\_equal(a=-49, b=-99, c=-56)

4 if -49 != 0: --- True

5 D = -99\*-99 - 4\*-49\*-56

D = -1175

6 if -1175 > 0: --- False

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

12 else:

13 return []

6. findmax([-79, 95, 13, -4]) = 95

1 def findmax(items=[-79, 95, 13, -4])

2 if len([-79, 95, 13, -4]) == 0: --- False

4 m = items[0]

m = -79

5 i = 1

6 while 1 < len([-79, 95, 13, -4]): --- True

7 if -79 < items[1]: --- True

8 m = items[1]

m = 95

9 i = 1 + 1

i = 2

6 while 2 < len([-79, 95, 13, -4]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([-79, 95, 13, -4]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([-79, 95, 13, -4]): --- False

10 return 95

7. unique([14, 14, -97]) = [14, -97]

1 def unique(items=[14, 14, -97])

2 res = []

3 i = 0

4 while 0 < len([14, 14, -97]): --- 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, 14, -97]): --- True

5 if len([14]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([14, 14, -97]): --- True

5 if len([14]) == 0 or res[-1] != items[2]: --- True

6 res = [14] + [items[2]]

res = [14, -97]

7 i = 2 + 1

i = 3

4 while 3 < len([14, 14, -97]): --- False

8 return [14, -97]

8. join('+', [80, 17, 84, 21]) = '80+17+84+21'

1 def join(sep='+', items=[80, 17, 84, 21])

2 res = ''

3 if len([80, 17, 84, 21]) > 0: --- True

4 res = str(items[0])

res = '80'

5 items = items[1:]

items = [17, 84, 21]

6 while len([17, 84, 21]) > 0: --- True

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

res = '80+17'

8 items = items[1:]

items = [84, 21]

6 while len([84, 21]) > 0: --- True

7 res = '80+17' + '+' + str(items[0])

res = '80+17+84'

8 items = items[1:]

items = [21]

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

7 res = '80+17+84' + '+' + str(items[0])

res = '80+17+84+21'

8 items = items[1:]

items = []

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

9 return '80+17+84+21'

Вариант: 2-4-11

1. gcd(-72, 76) = 4

1 def gcd(x=-72, y=76)

2 if -72 < 0: --- True

3 x = --72

x = 72

4 if 76 < 0: --- False

6 if 72 == 0: --- False

8 while 76 != 0: --- True

9 rem = 72 % 76

rem = 72

10 x = 76

11 y = 72

8 while 72 != 0: --- True

9 rem = 76 % 72

rem = 4

10 x = 72

11 y = 4

8 while 4 != 0: --- True

9 rem = 72 % 4

rem = 0

10 x = 4

11 y = 0

8 while 0 != 0: --- False

12 return 4

2. gcd(69, 0) = 69

1 def gcd(x=69, y=0)

2 if 69 < 0: --- False

4 if 0 < 0: --- False

6 if 69 == 0: --- False

8 while 0 != 0: --- False

12 return 69

3. hex(212) = 'D4'

3 def hex(number=212)

4 if 212 == 0: --- False

6 res = ''

7 while 212 > 0: --- True

8 digit = 212 % 16

digit = 4

9 res = DIGITS[4] + ''

res = '4'

10 number = 212 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + '4'

res = 'D4'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'D4'

4. square\_equal(-20, 3, 74) = [2.0, -1.85]

3 def square\_equal(a=-20, b=3, c=74)

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

5 D = 3\*3 - 4\*-20\*74

D = 5929

6 if 5929 > 0: --- True

7 x1 = (-3 - sqrt(5929)) / (2\*-20)

x1 = 2.0

8 x2 = (-3 + sqrt(5929)) / (2\*-20)

x2 = -1.85

9 return [2.0, -1.85]

5. square\_equal(30, -76, 78) = []

3 def square\_equal(a=30, b=-76, c=78)

4 if 30 != 0: --- True

5 D = -76\*-76 - 4\*30\*78

D = -3584

6 if -3584 > 0: --- False

10 elif -3584 == 0: --- False

12 else:

13 return []

6. findmax([69, -75, 15, 54, 28, -89]) = 69

1 def findmax(items=[69, -75, 15, 54, 28, -89])

2 if len([69, -75, 15, 54, 28, -89]) == 0: --- False

4 m = items[0]

m = 69

5 i = 1

6 while 1 < len([69, -75, 15, 54, 28, -89]): --- True

7 if 69 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([69, -75, 15, 54, 28, -89]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([69, -75, 15, 54, 28, -89]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([69, -75, 15, 54, 28, -89]): --- True

7 if 69 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([69, -75, 15, 54, 28, -89]): --- True

7 if 69 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([69, -75, 15, 54, 28, -89]): --- False

10 return 69

7. unique([-50, -92, -50, -50]) = [-50, -92, -50]

1 def unique(items=[-50, -92, -50, -50])

2 res = []

3 i = 0

4 while 0 < len([-50, -92, -50, -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, -92, -50, -50]): --- True

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

6 res = [-50] + [items[1]]

res = [-50, -92]

7 i = 1 + 1

i = 2

4 while 2 < len([-50, -92, -50, -50]): --- True

5 if len([-50, -92]) == 0 or res[-1] != items[2]: --- True

6 res = [-50, -92] + [items[2]]

res = [-50, -92, -50]

7 i = 2 + 1

i = 3

4 while 3 < len([-50, -92, -50, -50]): --- True

5 if len([-50, -92, -50]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([-50, -92, -50, -50]): --- False

8 return [-50, -92, -50]

8. join(':', [75, 85, 1]) = '75:85:1'

1 def join(sep=':', items=[75, 85, 1])

2 res = ''

3 if len([75, 85, 1]) > 0: --- True

4 res = str(items[0])

res = '75'

5 items = items[1:]

items = [85, 1]

6 while len([85, 1]) > 0: --- True

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

res = '75:85'

8 items = items[1:]

items = [1]

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

7 res = '75:85' + ':' + str(items[0])

res = '75:85:1'

8 items = items[1:]

items = []

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

9 return '75:85:1'

Вариант: 2-4-12

1. gcd(87, 18) = 3

1 def gcd(x=87, y=18)

2 if 87 < 0: --- False

4 if 18 < 0: --- False

6 if 87 == 0: --- False

8 while 18 != 0: --- True

9 rem = 87 % 18

rem = 15

10 x = 18

11 y = 15

8 while 15 != 0: --- True

9 rem = 18 % 15

rem = 3

10 x = 15

11 y = 3

8 while 3 != 0: --- True

9 rem = 15 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(63, 0) = 63

1 def gcd(x=63, y=0)

2 if 63 < 0: --- False

4 if 0 < 0: --- False

6 if 63 == 0: --- False

8 while 0 != 0: --- False

12 return 63

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(88, -22, 0) = [0.0, 0.25]

3 def square\_equal(a=88, b=-22, c=0)

4 if 88 != 0: --- True

5 D = -22\*-22 - 4\*88\*0

D = 484

6 if 484 > 0: --- True

7 x1 = (--22 - sqrt(484)) / (2\*88)

x1 = 0.0

8 x2 = (--22 + sqrt(484)) / (2\*88)

x2 = 0.25

9 return [0.0, 0.25]

5. square\_equal(-78, -8, -34) = []

3 def square\_equal(a=-78, b=-8, c=-34)

4 if -78 != 0: --- True

5 D = -8\*-8 - 4\*-78\*-34

D = -10544

6 if -10544 > 0: --- False

10 elif -10544 == 0: --- False

12 else:

13 return []

6. findmax([97, -42, 26, -63, 30]) = 97

1 def findmax(items=[97, -42, 26, -63, 30])

2 if len([97, -42, 26, -63, 30]) == 0: --- False

4 m = items[0]

m = 97

5 i = 1

6 while 1 < len([97, -42, 26, -63, 30]): --- True

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

9 i = 1 + 1

i = 2

6 while 2 < len([97, -42, 26, -63, 30]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([97, -42, 26, -63, 30]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([97, -42, 26, -63, 30]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([97, -42, 26, -63, 30]): --- False

10 return 97

7. unique([-17, 59, 59, -62]) = [-17, 59, -62]

1 def unique(items=[-17, 59, 59, -62])

2 res = []

3 i = 0

4 while 0 < len([-17, 59, 59, -62]): --- True

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

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

res = [-17]

7 i = 0 + 1

i = 1

4 while 1 < len([-17, 59, 59, -62]): --- True

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

6 res = [-17] + [items[1]]

res = [-17, 59]

7 i = 1 + 1

i = 2

4 while 2 < len([-17, 59, 59, -62]): --- True

5 if len([-17, 59]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-17, 59, 59, -62]): --- True

5 if len([-17, 59]) == 0 or res[-1] != items[3]: --- True

6 res = [-17, 59] + [items[3]]

res = [-17, 59, -62]

7 i = 3 + 1

i = 4

4 while 4 < len([-17, 59, 59, -62]): --- False

8 return [-17, 59, -62]

8. join(',', [89, 22, 91, 12]) = '89,22,91,12'

1 def join(sep=',', items=[89, 22, 91, 12])

2 res = ''

3 if len([89, 22, 91, 12]) > 0: --- True

4 res = str(items[0])

res = '89'

5 items = items[1:]

items = [22, 91, 12]

6 while len([22, 91, 12]) > 0: --- True

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

res = '89,22'

8 items = items[1:]

items = [91, 12]

6 while len([91, 12]) > 0: --- True

7 res = '89,22' + ',' + str(items[0])

res = '89,22,91'

8 items = items[1:]

items = [12]

6 while len([12]) > 0: --- True

7 res = '89,22,91' + ',' + str(items[0])

res = '89,22,91,12'

8 items = items[1:]

items = []

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

9 return '89,22,91,12'

Вариант: 2-4-13

1. gcd(52, -78) = 26

1 def gcd(x=52, y=-78)

2 if 52 < 0: --- False

4 if -78 < 0: --- True

5 y = --78

y = 78

6 if 52 == 0: --- False

8 while 78 != 0: --- True

9 rem = 52 % 78

rem = 52

10 x = 78

11 y = 52

8 while 52 != 0: --- True

9 rem = 78 % 52

rem = 26

10 x = 52

11 y = 26

8 while 26 != 0: --- True

9 rem = 52 % 26

rem = 0

10 x = 26

11 y = 0

8 while 0 != 0: --- False

12 return 26

2. gcd(0, 50) = 50

1 def gcd(x=0, y=50)

2 if 0 < 0: --- False

4 if 50 < 0: --- False

6 if 0 == 0: --- True

7 return 50

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(2, -7, -9) = [-1.0, 4.5]

3 def square\_equal(a=2, b=-7, c=-9)

4 if 2 != 0: --- True

5 D = -7\*-7 - 4\*2\*-9

D = 121

6 if 121 > 0: --- True

7 x1 = (--7 - sqrt(121)) / (2\*2)

x1 = -1.0

8 x2 = (--7 + sqrt(121)) / (2\*2)

x2 = 4.5

9 return [-1.0, 4.5]

5. square\_equal(54, 97, 54) = []

3 def square\_equal(a=54, b=97, c=54)

4 if 54 != 0: --- True

5 D = 97\*97 - 4\*54\*54

D = -2255

6 if -2255 > 0: --- False

10 elif -2255 == 0: --- False

12 else:

13 return []

6. findmax([65, 41, 80, 39]) = 80

1 def findmax(items=[65, 41, 80, 39])

2 if len([65, 41, 80, 39]) == 0: --- False

4 m = items[0]

m = 65

5 i = 1

6 while 1 < len([65, 41, 80, 39]): --- True

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

9 i = 1 + 1

i = 2

6 while 2 < len([65, 41, 80, 39]): --- True

7 if 65 < items[2]: --- True

8 m = items[2]

m = 80

9 i = 2 + 1

i = 3

6 while 3 < len([65, 41, 80, 39]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([65, 41, 80, 39]): --- False

10 return 80

7. unique([-78, -77, -77, -86]) = [-78, -77, -86]

1 def unique(items=[-78, -77, -77, -86])

2 res = []

3 i = 0

4 while 0 < len([-78, -77, -77, -86]): --- 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, -77, -77, -86]): --- True

5 if len([-78]) == 0 or res[-1] != items[1]: --- True

6 res = [-78] + [items[1]]

res = [-78, -77]

7 i = 1 + 1

i = 2

4 while 2 < len([-78, -77, -77, -86]): --- True

5 if len([-78, -77]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-78, -77, -77, -86]): --- True

5 if len([-78, -77]) == 0 or res[-1] != items[3]: --- True

6 res = [-78, -77] + [items[3]]

res = [-78, -77, -86]

7 i = 3 + 1

i = 4

4 while 4 < len([-78, -77, -77, -86]): --- False

8 return [-78, -77, -86]

8. join(';', [40, 17, 26, 34]) = '40;17;26;34'

1 def join(sep=';', items=[40, 17, 26, 34])

2 res = ''

3 if len([40, 17, 26, 34]) > 0: --- True

4 res = str(items[0])

res = '40'

5 items = items[1:]

items = [17, 26, 34]

6 while len([17, 26, 34]) > 0: --- True

7 res = '40' + ';' + str(items[0])

res = '40;17'

8 items = items[1:]

items = [26, 34]

6 while len([26, 34]) > 0: --- True

7 res = '40;17' + ';' + str(items[0])

res = '40;17;26'

8 items = items[1:]

items = [34]

6 while len([34]) > 0: --- True

7 res = '40;17;26' + ';' + str(items[0])

res = '40;17;26;34'

8 items = items[1:]

items = []

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

9 return '40;17;26;34'

Вариант: 2-4-14

1. gcd(-20, 35) = 5

1 def gcd(x=-20, y=35)

2 if -20 < 0: --- True

3 x = --20

x = 20

4 if 35 < 0: --- False

6 if 20 == 0: --- False

8 while 35 != 0: --- True

9 rem = 20 % 35

rem = 20

10 x = 35

11 y = 20

8 while 20 != 0: --- True

9 rem = 35 % 20

rem = 15

10 x = 20

11 y = 15

8 while 15 != 0: --- True

9 rem = 20 % 15

rem = 5

10 x = 15

11 y = 5

8 while 5 != 0: --- True

9 rem = 15 % 5

rem = 0

10 x = 5

11 y = 0

8 while 0 != 0: --- False

12 return 5

2. gcd(0, 19) = 19

1 def gcd(x=0, y=19)

2 if 0 < 0: --- False

4 if 19 < 0: --- False

6 if 0 == 0: --- True

7 return 19

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(7, -56, 49) = [1.0, 7.0]

3 def square\_equal(a=7, b=-56, c=49)

4 if 7 != 0: --- True

5 D = -56\*-56 - 4\*7\*49

D = 1764

6 if 1764 > 0: --- True

7 x1 = (--56 - sqrt(1764)) / (2\*7)

x1 = 1.0

8 x2 = (--56 + sqrt(1764)) / (2\*7)

x2 = 7.0

9 return [1.0, 7.0]

5. square\_equal(70, 12, 57) = []

3 def square\_equal(a=70, b=12, c=57)

4 if 70 != 0: --- True

5 D = 12\*12 - 4\*70\*57

D = -15816

6 if -15816 > 0: --- False

10 elif -15816 == 0: --- False

12 else:

13 return []

6. findmax([26, 84, 28, -66]) = 84

1 def findmax(items=[26, 84, 28, -66])

2 if len([26, 84, 28, -66]) == 0: --- False

4 m = items[0]

m = 26

5 i = 1

6 while 1 < len([26, 84, 28, -66]): --- True

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

8 m = items[1]

m = 84

9 i = 1 + 1

i = 2

6 while 2 < len([26, 84, 28, -66]): --- True

7 if 84 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([26, 84, 28, -66]): --- True

7 if 84 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([26, 84, 28, -66]): --- False

10 return 84

7. unique([62, 62, -2, 62]) = [62, -2, 62]

1 def unique(items=[62, 62, -2, 62])

2 res = []

3 i = 0

4 while 0 < len([62, 62, -2, 62]): --- True

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

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

res = [62]

7 i = 0 + 1

i = 1

4 while 1 < len([62, 62, -2, 62]): --- True

5 if len([62]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([62, 62, -2, 62]): --- True

5 if len([62]) == 0 or res[-1] != items[2]: --- True

6 res = [62] + [items[2]]

res = [62, -2]

7 i = 2 + 1

i = 3

4 while 3 < len([62, 62, -2, 62]): --- True

5 if len([62, -2]) == 0 or res[-1] != items[3]: --- True

6 res = [62, -2] + [items[3]]

res = [62, -2, 62]

7 i = 3 + 1

i = 4

4 while 4 < len([62, 62, -2, 62]): --- False

8 return [62, -2, 62]

8. join(';', [64, 35, 42]) = '64;35;42'

1 def join(sep=';', items=[64, 35, 42])

2 res = ''

3 if len([64, 35, 42]) > 0: --- True

4 res = str(items[0])

res = '64'

5 items = items[1:]

items = [35, 42]

6 while len([35, 42]) > 0: --- True

7 res = '64' + ';' + str(items[0])

res = '64;35'

8 items = items[1:]

items = [42]

6 while len([42]) > 0: --- True

7 res = '64;35' + ';' + str(items[0])

res = '64;35;42'

8 items = items[1:]

items = []

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

9 return '64;35;42'

Вариант: 2-4-15

1. gcd(-15, -63) = 3

1 def gcd(x=-15, y=-63)

2 if -15 < 0: --- True

3 x = --15

x = 15

4 if -63 < 0: --- True

5 y = --63

y = 63

6 if 15 == 0: --- False

8 while 63 != 0: --- True

9 rem = 15 % 63

rem = 15

10 x = 63

11 y = 15

8 while 15 != 0: --- True

9 rem = 63 % 15

rem = 3

10 x = 15

11 y = 3

8 while 3 != 0: --- True

9 rem = 15 % 3

rem = 0

10 x = 3

11 y = 0

8 while 0 != 0: --- False

12 return 3

2. gcd(-49, 0) = 49

1 def gcd(x=-49, y=0)

2 if -49 < 0: --- True

3 x = --49

x = 49

4 if 0 < 0: --- False

6 if 49 == 0: --- False

8 while 0 != 0: --- False

12 return 49

3. hex(208) = 'D0'

3 def hex(number=208)

4 if 208 == 0: --- False

6 res = ''

7 while 208 > 0: --- True

8 digit = 208 % 16

digit = 0

9 res = DIGITS[0] + ''

res = '0'

10 number = 208 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + '0'

res = 'D0'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'D0'

4. square\_equal(0, -14, 63) = [4.5]

3 def square\_equal(a=0, b=-14, c=63)

4 if 0 != 0: --- False

14 else:

15 if -14 != 0: --- True

16 return [4.5]

5. square\_equal(56, -15, 88) = []

3 def square\_equal(a=56, b=-15, c=88)

4 if 56 != 0: --- True

5 D = -15\*-15 - 4\*56\*88

D = -19487

6 if -19487 > 0: --- False

10 elif -19487 == 0: --- False

12 else:

13 return []

6. findmax([-22, 14, 14, 98, -38]) = 98

1 def findmax(items=[-22, 14, 14, 98, -38])

2 if len([-22, 14, 14, 98, -38]) == 0: --- False

4 m = items[0]

m = -22

5 i = 1

6 while 1 < len([-22, 14, 14, 98, -38]): --- True

7 if -22 < items[1]: --- True

8 m = items[1]

m = 14

9 i = 1 + 1

i = 2

6 while 2 < len([-22, 14, 14, 98, -38]): --- True

7 if 14 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-22, 14, 14, 98, -38]): --- True

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

8 m = items[3]

m = 98

9 i = 3 + 1

i = 4

6 while 4 < len([-22, 14, 14, 98, -38]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([-22, 14, 14, 98, -38]): --- False

10 return 98

7. unique([-49, -28, -28]) = [-49, -28]

1 def unique(items=[-49, -28, -28])

2 res = []

3 i = 0

4 while 0 < len([-49, -28, -28]): --- True

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

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

res = [-49]

7 i = 0 + 1

i = 1

4 while 1 < len([-49, -28, -28]): --- True

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

6 res = [-49] + [items[1]]

res = [-49, -28]

7 i = 1 + 1

i = 2

4 while 2 < len([-49, -28, -28]): --- True

5 if len([-49, -28]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([-49, -28, -28]): --- False

8 return [-49, -28]

8. join(':', [56, 3, 63]) = '56:3:63'

1 def join(sep=':', items=[56, 3, 63])

2 res = ''

3 if len([56, 3, 63]) > 0: --- True

4 res = str(items[0])

res = '56'

5 items = items[1:]

items = [3, 63]

6 while len([3, 63]) > 0: --- True

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

res = '56:3'

8 items = items[1:]

items = [63]

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

7 res = '56:3' + ':' + str(items[0])

res = '56:3:63'

8 items = items[1:]

items = []

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

9 return '56:3:63'

Вариант: 2-4-16

1. gcd(-50, -100) = 50

1 def gcd(x=-50, y=-100)

2 if -50 < 0: --- True

3 x = --50

x = 50

4 if -100 < 0: --- True

5 y = --100

y = 100

6 if 50 == 0: --- False

8 while 100 != 0: --- True

9 rem = 50 % 100

rem = 50

10 x = 100

11 y = 50

8 while 50 != 0: --- True

9 rem = 100 % 50

rem = 0

10 x = 50

11 y = 0

8 while 0 != 0: --- False

12 return 50

2. gcd(-84, 0) = 84

1 def gcd(x=-84, y=0)

2 if -84 < 0: --- True

3 x = --84

x = 84

4 if 0 < 0: --- False

6 if 84 == 0: --- False

8 while 0 != 0: --- False

12 return 84

3. hex(182) = 'B6'

3 def hex(number=182)

4 if 182 == 0: --- False

6 res = ''

7 while 182 > 0: --- True

8 digit = 182 % 16

digit = 6

9 res = DIGITS[6] + ''

res = '6'

10 number = 182 // 16

number = 11

7 while 11 > 0: --- True

8 digit = 11 % 16

digit = 11

9 res = DIGITS[11] + '6'

res = 'B6'

10 number = 11 // 16

number = 0

7 while 0 > 0: --- False

11 return 'B6'

4. square\_equal(-26, 65, 39) = [3.0, -0.5]

3 def square\_equal(a=-26, b=65, c=39)

4 if -26 != 0: --- True

5 D = 65\*65 - 4\*-26\*39

D = 8281

6 if 8281 > 0: --- True

7 x1 = (-65 - sqrt(8281)) / (2\*-26)

x1 = 3.0

8 x2 = (-65 + sqrt(8281)) / (2\*-26)

x2 = -0.5

9 return [3.0, -0.5]

5. square\_equal(75, 64, 25) = []

3 def square\_equal(a=75, b=64, c=25)

4 if 75 != 0: --- True

5 D = 64\*64 - 4\*75\*25

D = -3404

6 if -3404 > 0: --- False

10 elif -3404 == 0: --- False

12 else:

13 return []

6. findmax([-2, 12, 80, -27, 74]) = 80

1 def findmax(items=[-2, 12, 80, -27, 74])

2 if len([-2, 12, 80, -27, 74]) == 0: --- False

4 m = items[0]

m = -2

5 i = 1

6 while 1 < len([-2, 12, 80, -27, 74]): --- True

7 if -2 < items[1]: --- True

8 m = items[1]

m = 12

9 i = 1 + 1

i = 2

6 while 2 < len([-2, 12, 80, -27, 74]): --- True

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

8 m = items[2]

m = 80

9 i = 2 + 1

i = 3

6 while 3 < len([-2, 12, 80, -27, 74]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([-2, 12, 80, -27, 74]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([-2, 12, 80, -27, 74]): --- False

10 return 80

7. unique([28, -96, -96, 70]) = [28, -96, 70]

1 def unique(items=[28, -96, -96, 70])

2 res = []

3 i = 0

4 while 0 < len([28, -96, -96, 70]): --- True

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

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

res = [28]

7 i = 0 + 1

i = 1

4 while 1 < len([28, -96, -96, 70]): --- True

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

6 res = [28] + [items[1]]

res = [28, -96]

7 i = 1 + 1

i = 2

4 while 2 < len([28, -96, -96, 70]): --- True

5 if len([28, -96]) == 0 or res[-1] != items[2]: --- False

7 i = 2 + 1

i = 3

4 while 3 < len([28, -96, -96, 70]): --- True

5 if len([28, -96]) == 0 or res[-1] != items[3]: --- True

6 res = [28, -96] + [items[3]]

res = [28, -96, 70]

7 i = 3 + 1

i = 4

4 while 4 < len([28, -96, -96, 70]): --- False

8 return [28, -96, 70]

8. join(',', [29, 89, 50]) = '29,89,50'

1 def join(sep=',', items=[29, 89, 50])

2 res = ''

3 if len([29, 89, 50]) > 0: --- True

4 res = str(items[0])

res = '29'

5 items = items[1:]

items = [89, 50]

6 while len([89, 50]) > 0: --- True

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

res = '29,89'

8 items = items[1:]

items = [50]

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

7 res = '29,89' + ',' + str(items[0])

res = '29,89,50'

8 items = items[1:]

items = []

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

9 return '29,89,50'

Вариант: 2-4-17

1. gcd(72, -20) = 4

1 def gcd(x=72, y=-20)

2 if 72 < 0: --- False

4 if -20 < 0: --- True

5 y = --20

y = 20

6 if 72 == 0: --- False

8 while 20 != 0: --- True

9 rem = 72 % 20

rem = 12

10 x = 20

11 y = 12

8 while 12 != 0: --- True

9 rem = 20 % 12

rem = 8

10 x = 12

11 y = 8

8 while 8 != 0: --- True

9 rem = 12 % 8

rem = 4

10 x = 8

11 y = 4

8 while 4 != 0: --- True

9 rem = 8 % 4

rem = 0

10 x = 4

11 y = 0

8 while 0 != 0: --- False

12 return 4

2. gcd(32, 0) = 32

1 def gcd(x=32, y=0)

2 if 32 < 0: --- False

4 if 0 < 0: --- False

6 if 32 == 0: --- False

8 while 0 != 0: --- False

12 return 32

3. hex(210) = 'D2'

3 def hex(number=210)

4 if 210 == 0: --- False

6 res = ''

7 while 210 > 0: --- True

8 digit = 210 % 16

digit = 2

9 res = DIGITS[2] + ''

res = '2'

10 number = 210 // 16

number = 13

7 while 13 > 0: --- True

8 digit = 13 % 16

digit = 13

9 res = DIGITS[13] + '2'

res = 'D2'

10 number = 13 // 16

number = 0

7 while 0 > 0: --- False

11 return 'D2'

4. square\_equal(-4, -47, 12) = [0.25, -12.0]

3 def square\_equal(a=-4, b=-47, c=12)

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

5 D = -47\*-47 - 4\*-4\*12

D = 2401

6 if 2401 > 0: --- True

7 x1 = (--47 - sqrt(2401)) / (2\*-4)

x1 = 0.25

8 x2 = (--47 + sqrt(2401)) / (2\*-4)

x2 = -12.0

9 return [0.25, -12.0]

5. square\_equal(67, 55, 44) = []

3 def square\_equal(a=67, b=55, c=44)

4 if 67 != 0: --- True

5 D = 55\*55 - 4\*67\*44

D = -8767

6 if -8767 > 0: --- False

10 elif -8767 == 0: --- False

12 else:

13 return []

6. findmax([93, -8, -58, -38, 71]) = 93

1 def findmax(items=[93, -8, -58, -38, 71])

2 if len([93, -8, -58, -38, 71]) == 0: --- False

4 m = items[0]

m = 93

5 i = 1

6 while 1 < len([93, -8, -58, -38, 71]): --- True

7 if 93 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([93, -8, -58, -38, 71]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([93, -8, -58, -38, 71]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([93, -8, -58, -38, 71]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([93, -8, -58, -38, 71]): --- False

10 return 93

7. unique([-92, -93, -92, -92]) = [-92, -93, -92]

1 def unique(items=[-92, -93, -92, -92])

2 res = []

3 i = 0

4 while 0 < len([-92, -93, -92, -92]): --- True

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

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

res = [-92]

7 i = 0 + 1

i = 1

4 while 1 < len([-92, -93, -92, -92]): --- True

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

6 res = [-92] + [items[1]]

res = [-92, -93]

7 i = 1 + 1

i = 2

4 while 2 < len([-92, -93, -92, -92]): --- True

5 if len([-92, -93]) == 0 or res[-1] != items[2]: --- True

6 res = [-92, -93] + [items[2]]

res = [-92, -93, -92]

7 i = 2 + 1

i = 3

4 while 3 < len([-92, -93, -92, -92]): --- True

5 if len([-92, -93, -92]) == 0 or res[-1] != items[3]: --- False

7 i = 3 + 1

i = 4

4 while 4 < len([-92, -93, -92, -92]): --- False

8 return [-92, -93, -92]

8. join('+', [15, 91, 9, 31]) = '15+91+9+31'

1 def join(sep='+', items=[15, 91, 9, 31])

2 res = ''

3 if len([15, 91, 9, 31]) > 0: --- True

4 res = str(items[0])

res = '15'

5 items = items[1:]

items = [91, 9, 31]

6 while len([91, 9, 31]) > 0: --- True

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

res = '15+91'

8 items = items[1:]

items = [9, 31]

6 while len([9, 31]) > 0: --- True

7 res = '15+91' + '+' + str(items[0])

res = '15+91+9'

8 items = items[1:]

items = [31]

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

7 res = '15+91+9' + '+' + str(items[0])

res = '15+91+9+31'

8 items = items[1:]

items = []

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

9 return '15+91+9+31'

Вариант: 2-4-18

1. gcd(-88, 55) = 11

1 def gcd(x=-88, y=55)

2 if -88 < 0: --- True

3 x = --88

x = 88

4 if 55 < 0: --- False

6 if 88 == 0: --- False

8 while 55 != 0: --- True

9 rem = 88 % 55

rem = 33

10 x = 55

11 y = 33

8 while 33 != 0: --- True

9 rem = 55 % 33

rem = 22

10 x = 33

11 y = 22

8 while 22 != 0: --- True

9 rem = 33 % 22

rem = 11

10 x = 22

11 y = 11

8 while 11 != 0: --- True

9 rem = 22 % 11

rem = 0

10 x = 11

11 y = 0

8 while 0 != 0: --- False

12 return 11

2. gcd(0, -79) = 79

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

2 if 0 < 0: --- False

4 if -79 < 0: --- True

5 y = --79

y = 79

6 if 0 == 0: --- True

7 return 79

3. hex(249) = 'F9'

3 def hex(number=249)

4 if 249 == 0: --- False

6 res = ''

7 while 249 > 0: --- True

8 digit = 249 % 16

digit = 9

9 res = DIGITS[9] + ''

res = '9'

10 number = 249 // 16

number = 15

7 while 15 > 0: --- True

8 digit = 15 % 16

digit = 15

9 res = DIGITS[15] + '9'

res = 'F9'

10 number = 15 // 16

number = 0

7 while 0 > 0: --- False

11 return 'F9'

4. square\_equal(-15, -33, -6) = [-0.2, -2.0]

3 def square\_equal(a=-15, b=-33, c=-6)

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

5 D = -33\*-33 - 4\*-15\*-6

D = 729

6 if 729 > 0: --- True

7 x1 = (--33 - sqrt(729)) / (2\*-15)

x1 = -0.2

8 x2 = (--33 + sqrt(729)) / (2\*-15)

x2 = -2.0

9 return [-0.2, -2.0]

5. square\_equal(-31, -28, -12) = []

3 def square\_equal(a=-31, b=-28, c=-12)

4 if -31 != 0: --- True

5 D = -28\*-28 - 4\*-31\*-12

D = -704

6 if -704 > 0: --- False

10 elif -704 == 0: --- False

12 else:

13 return []

6. findmax([74, -94, -77, -92, -85, 1]) = 74

1 def findmax(items=[74, -94, -77, -92, -85, 1])

2 if len([74, -94, -77, -92, -85, 1]) == 0: --- False

4 m = items[0]

m = 74

5 i = 1

6 while 1 < len([74, -94, -77, -92, -85, 1]): --- True

7 if 74 < items[1]: --- False

9 i = 1 + 1

i = 2

6 while 2 < len([74, -94, -77, -92, -85, 1]): --- True

7 if 74 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([74, -94, -77, -92, -85, 1]): --- True

7 if 74 < items[3]: --- False

9 i = 3 + 1

i = 4

6 while 4 < len([74, -94, -77, -92, -85, 1]): --- True

7 if 74 < items[4]: --- False

9 i = 4 + 1

i = 5

6 while 5 < len([74, -94, -77, -92, -85, 1]): --- True

7 if 74 < items[5]: --- False

9 i = 5 + 1

i = 6

6 while 6 < len([74, -94, -77, -92, -85, 1]): --- False

10 return 74

7. unique([35, 35, -54]) = [35, -54]

1 def unique(items=[35, 35, -54])

2 res = []

3 i = 0

4 while 0 < len([35, 35, -54]): --- True

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

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

res = [35]

7 i = 0 + 1

i = 1

4 while 1 < len([35, 35, -54]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([35, 35, -54]): --- True

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

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

res = [35, -54]

7 i = 2 + 1

i = 3

4 while 3 < len([35, 35, -54]): --- False

8 return [35, -54]

8. join(':', [19, 74, 19, 16]) = '19:74:19:16'

1 def join(sep=':', items=[19, 74, 19, 16])

2 res = ''

3 if len([19, 74, 19, 16]) > 0: --- True

4 res = str(items[0])

res = '19'

5 items = items[1:]

items = [74, 19, 16]

6 while len([74, 19, 16]) > 0: --- True

7 res = '19' + ':' + str(items[0])

res = '19:74'

8 items = items[1:]

items = [19, 16]

6 while len([19, 16]) > 0: --- True

7 res = '19:74' + ':' + str(items[0])

res = '19:74:19'

8 items = items[1:]

items = [16]

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

7 res = '19:74:19' + ':' + str(items[0])

res = '19:74:19:16'

8 items = items[1:]

items = []

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

9 return '19:74:19:16'

Вариант: 2-4-19

1. gcd(-21, 24) = 3

1 def gcd(x=-21, y=24)

2 if -21 < 0: --- True

3 x = --21

x = 21

4 if 24 < 0: --- False

6 if 21 == 0: --- False

8 while 24 != 0: --- True

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

1 def gcd(x=0, y=9)

2 if 0 < 0: --- False

4 if 9 < 0: --- False

6 if 0 == 0: --- True

7 return 9

3. hex(236) = 'EC'

3 def hex(number=236)

4 if 236 == 0: --- False

6 res = ''

7 while 236 > 0: --- True

8 digit = 236 % 16

digit = 12

9 res = DIGITS[12] + ''

res = 'C'

10 number = 236 // 16

number = 14

7 while 14 > 0: --- True

8 digit = 14 % 16

digit = 14

9 res = DIGITS[14] + 'C'

res = 'EC'

10 number = 14 // 16

number = 0

7 while 0 > 0: --- False

11 return 'EC'

4. square\_equal(4, -41, 100) = [4.0, 6.25]

3 def square\_equal(a=4, b=-41, c=100)

4 if 4 != 0: --- True

5 D = -41\*-41 - 4\*4\*100

D = 81

6 if 81 > 0: --- True

7 x1 = (--41 - sqrt(81)) / (2\*4)

x1 = 4.0

8 x2 = (--41 + sqrt(81)) / (2\*4)

x2 = 6.25

9 return [4.0, 6.25]

5. square\_equal(-44, 17, -11) = []

3 def square\_equal(a=-44, b=17, c=-11)

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

5 D = 17\*17 - 4\*-44\*-11

D = -1647

6 if -1647 > 0: --- False

10 elif -1647 == 0: --- False

12 else:

13 return []

6. findmax([-4, 38, -67, 67, 77]) = 77

1 def findmax(items=[-4, 38, -67, 67, 77])

2 if len([-4, 38, -67, 67, 77]) == 0: --- False

4 m = items[0]

m = -4

5 i = 1

6 while 1 < len([-4, 38, -67, 67, 77]): --- True

7 if -4 < items[1]: --- True

8 m = items[1]

m = 38

9 i = 1 + 1

i = 2

6 while 2 < len([-4, 38, -67, 67, 77]): --- True

7 if 38 < items[2]: --- False

9 i = 2 + 1

i = 3

6 while 3 < len([-4, 38, -67, 67, 77]): --- True

7 if 38 < items[3]: --- True

8 m = items[3]

m = 67

9 i = 3 + 1

i = 4

6 while 4 < len([-4, 38, -67, 67, 77]): --- True

7 if 67 < items[4]: --- True

8 m = items[4]

m = 77

9 i = 4 + 1

i = 5

6 while 5 < len([-4, 38, -67, 67, 77]): --- False

10 return 77

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

1 def unique(items=[99, 99, 7, 99])

2 res = []

3 i = 0

4 while 0 < len([99, 99, 7, 99]): --- True

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

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

res = [99]

7 i = 0 + 1

i = 1

4 while 1 < len([99, 99, 7, 99]): --- True

5 if len([99]) == 0 or res[-1] != items[1]: --- False

7 i = 1 + 1

i = 2

4 while 2 < len([99, 99, 7, 99]): --- True

5 if len([99]) == 0 or res[-1] != items[2]: --- True

6 res = [99] + [items[2]]

res = [99, 7]

7 i = 2 + 1

i = 3

4 while 3 < len([99, 99, 7, 99]): --- True

5 if len([99, 7]) == 0 or res[-1] != items[3]: --- True

6 res = [99, 7] + [items[3]]

res = [99, 7, 99]

7 i = 3 + 1

i = 4

4 while 4 < len([99, 99, 7, 99]): --- False

8 return [99, 7, 99]

8. join(',', [89, 74, 83]) = '89,74,83'

1 def join(sep=',', items=[89, 74, 83])

2 res = ''

3 if len([89, 74, 83]) > 0: --- True

4 res = str(items[0])

res = '89'

5 items = items[1:]

items = [74, 83]

6 while len([74, 83]) > 0: --- True

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

res = '89,74'

8 items = items[1:]

items = [83]

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

7 res = '89,74' + ',' + str(items[0])

res = '89,74,83'

8 items = items[1:]

items = []

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

9 return '89,74,83'

Вариант: 2-4-20

1. gcd(100, 70) = 10

1 def gcd(x=100, y=70)

2 if 100 < 0: --- False

4 if 70 < 0: --- False

6 if 100 == 0: --- False

8 while 70 != 0: --- True

9 rem = 100 % 70

rem = 30

10 x = 70

11 y = 30

8 while 30 != 0: --- True

9 rem = 70 % 30

rem = 10

10 x = 30

11 y = 10

8 while 10 != 0: --- True

9 rem = 30 % 10

rem = 0

10 x = 10

11 y = 0

8 while 0 != 0: --- False

12 return 10

2. gcd(-14, 0) = 14

1 def gcd(x=-14, y=0)

2 if -14 < 0: --- True

3 x = --14

x = 14

4 if 0 < 0: --- False

6 if 14 == 0: --- False

8 while 0 != 0: --- False

12 return 14

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(-4, 23, -19) = [4.75, 1.0]

3 def square\_equal(a=-4, b=23, c=-19)

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

5 D = 23\*23 - 4\*-4\*-19

D = 225

6 if 225 > 0: --- True

7 x1 = (-23 - sqrt(225)) / (2\*-4)

x1 = 4.75

8 x2 = (-23 + sqrt(225)) / (2\*-4)

x2 = 1.0

9 return [4.75, 1.0]

5. square\_equal(12, -8, 75) = []

3 def square\_equal(a=12, b=-8, c=75)

4 if 12 != 0: --- True

5 D = -8\*-8 - 4\*12\*75

D = -3536

6 if -3536 > 0: --- False

10 elif -3536 == 0: --- False

12 else:

13 return []

6. findmax([-40, 43, 20, -29, -18]) = 43

1 def findmax(items=[-40, 43, 20, -29, -18])

2 if len([-40, 43, 20, -29, -18]) == 0: --- False

4 m = items[0]

m = -40

5 i = 1

6 while 1 < len([-40, 43, 20, -29, -18]): --- True

7 if -40 < items[1]: --- True

8 m = items[1]

m = 43

9 i = 1 + 1

i = 2

6 while 2 < len([-40, 43, 20, -29, -18]): --- True

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

9 i = 2 + 1

i = 3

6 while 3 < len([-40, 43, 20, -29, -18]): --- True

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

9 i = 3 + 1

i = 4

6 while 4 < len([-40, 43, 20, -29, -18]): --- True

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

9 i = 4 + 1

i = 5

6 while 5 < len([-40, 43, 20, -29, -18]): --- False

10 return 43

7. unique([63, 63, -64]) = [63, -64]

1 def unique(items=[63, 63, -64])

2 res = []

3 i = 0

4 while 0 < len([63, 63, -64]): --- True

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

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

res = [63]

7 i = 0 + 1

i = 1

4 while 1 < len([63, 63, -64]): --- True

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

7 i = 1 + 1

i = 2

4 while 2 < len([63, 63, -64]): --- True

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

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

res = [63, -64]

7 i = 2 + 1

i = 3

4 while 3 < len([63, 63, -64]): --- False

8 return [63, -64]

8. join('+', [98, 51, 80, 33]) = '98+51+80+33'

1 def join(sep='+', items=[98, 51, 80, 33])

2 res = ''

3 if len([98, 51, 80, 33]) > 0: --- True

4 res = str(items[0])

res = '98'

5 items = items[1:]

items = [51, 80, 33]

6 while len([51, 80, 33]) > 0: --- True

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

res = '98+51'

8 items = items[1:]

items = [80, 33]

6 while len([80, 33]) > 0: --- True

7 res = '98+51' + '+' + str(items[0])

res = '98+51+80'

8 items = items[1:]

items = [33]

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

7 res = '98+51+80' + '+' + str(items[0])

res = '98+51+80+33'

8 items = items[1:]

items = []

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

9 return '98+51+80+33'