NO:

DATE: / /

2

```
areAmicableNumbers(x, y):

# find x's and y's factors:
listofXfactors = [ ]
listofYfactors = [ ]
for num in range(1, x):
    if x % num == 0:
        listofXfactors.append(num)

for num2 in range(1, y):
    if y % num2 == 0:
        listofYfactors.append(num2)

if sum(listofXfactors) == Y
   & sum(listofYfactors) == X :
    return True

else:
    return False
```

NO:

DATE: 20' 06' 25.

#3. nthMarriedNumber(n)

```
def digitCount(n):
    return len(n).

def splitNumber(n, i):
    firstNum = (n // 10**i)
    secondNum = (n % 10**i)
    return (firstNum, secondNum)

def nthMarriedNumber(n):
    count = 0.
    testNum = 0.
    length of testNum = digitCount(n)
while count < n:
    for combinationNum in range(1, len…m):
        twoNums = splitNumber(n, combi…m)
        if areAmicableNumbers(twoNums[0],
                              twoNums[1]):
            count += 1
            break.

    testNum += 1

return testNum-1
```

Per-Duet

NO:

DATE: 20'06'25.

#4 runStringProgram(s)

Question: How to split string into data and commands?

```
s = "qPx YqP : S24.L?"
count = 0
countSplitData = 0.
# split string into data and command.
for char in s :
    if char == ":" :
        countSplitData = count.
        data = s[ : count]
        commandStr = s[count : ]
    count += 1
# Split command string into commands.
# 2 types of command : S & L.
# 2 types of S command : Sx. Sxy.

dataTempStr = " ".
commandS = " "

for charIdx in range(len(commandStr)):
    if commandStr[charIdx] == "S":
        if commandStr[charIdx + 3] == ".":
            dataTempStr = data[
                                commandStr[ch..x+1]
                                : commandStr[ch..x
                                +2]
                                ]
            dataTempStr = data[: com...r[
                                ch..x+1]
```

NO:

DATE: / /

```
if commandStr[charIdx] == "L":
    if dataTempStr are all lowercase:
        return True

~~if commandStr[charIdx] == "?":~~
    dataTempStr = ""

return False
```

Per-Duet

NO:

DATE: 20' 06' 25.

#5 Code Tracing.

```
        0 1 2 3 4 5 6 7
ctl("aB1C2xD3")
t = ' '
for i in range(8):
    if s[i].isupper():
        t += f(s[i:i+2])

    elif s[i].isdigit():
        t += s[i-1:i+2:2][::-1]

    else:
        t += '.\n'

return t
```

Order.

=> a: 97

A: 65

x: 120.

i=0 t=.\n

i=1 t += f(s[1:3])

→ t += f('B1') → chr(66+1)

→ t += '67'

→ t = .\n67

i=2. t += s[1:4:2][::-1]

→ t += 'BC'[::-1]

→ t += 'CB'

→ t = .\n67CB.

Per-Duet

NO:

DATE: 20'06'29

S6.2?, S13.2?

#5 Code Tracing:

```
def f(s):
    a = s[0]
    b = s[-1]
    return chr(ord(a) + int(b))
```

i=3 t += f(s[3:5])
→ t += f('C2')
→ t += chr(67+2)
→ t += '69'
→ t = '.\n67CB69'

i=4 t += s[3:6:2][::-1]
→ t += 'Cx'[::-1]
→ t += 'xC'
→ t = '.\n67CB69xC'

i=5 t += '.\n'
→ t = '.\n67CB69xC.\n'

i=6 t += f(s[6:8])
→ t += f('D3')
→ t += chr(68+3)
→ t += '71'

→ t = '.\n67CB69xC.\n71'

Per-Dust

NO:

DATE: / /

i=7. t += s[6:9:2][::-1]

→ t += 'D'[::-1]

→ t += 'D'

→ t = '.\n67CB69xC.\n71D'

⇒ Ans:

.

67CB69xC.

71D.

NO:

DATE: 20' 06' 25.

#7 Reasoning Over Code.

```
def f(n):
    a = 0
    while n > 0:
        n //= 10
        a += 1
    return a
```

⇒ a 代表這個數字是幾位數

```
def g(n):
    return int(n** 0.5) ** 2 == n
```

⇒ True 的條件為 $\sqrt{n}$ 為整數.

```
def r(n):
    assert(n > 99999)
    while (n > 0):
        a = n % 10
        b = n // 10 ** (f(n) - 1)
        (if not g(10 * b + a):
            return False)
        n //= 10
        n %= 10 ** (f(n) - 1)
    return True.
```

Per-Duet

NO:

DATE: / /

a: 代表 n 的個位數

b: 代表 n 的最高位數.

$10*b + a = 16$ $b=1$ $a=6$.

→ assume num = (10000b)

n // = 10

→ num = 10000

n % = 10 ** (f(n) - 1)

→ num = 0.

return True.