ЗМАГАННЯ

«ЦИКЛИ»

https://www.eolymp.com/uk/contests/6480

Задачі

A	Йо-йо
B	Рівні дільники
C	Ледар
D	Поставка содової води
E	Відношення добутку до суми
F	Дуже просто!!!
G	Кролики
H	Друга цифра числа
I	Число у зворотньому порядку
J	Кількість даних цифр в числі
K	Срібна медаль
L	Казка про яблуко
M	Відмінник Іванко

Задача (A) Йо-йо

Іграшка йо-йо складається з катушки, на яку намотано нитку. Якщо, тримаючи за кінець нитки, відпустити катушку, то вона буде, обертаючись, спочатку опускатись донизу, а потім за інерцією підніматись вверх. Але висота, на яку катушка підніметься, буде в k раз меншою, ніж висота, з якої вона опустилась. Будемо вважати, що катушка зупинилась, якщо висота її чергового підйому не перевищує 1.

Напишіть програму, яка за довжиною нитки l та коефіцієнту k рахує кількість підйомів катушки до зупинки. Наприклад, нехай l = 17, k = 2, тоді катушка буде підніматись на висоти 8.5, 4.25, 2.125, 1.0625, а потім зупиниться. Таким чином будемо мати 4 підйоми.

Вхідні дані
Два цілих числа l ($1 \le l \le 109$) та k ($2 \le k \le 100$).
Вихідні дані
Вивести одне число – кількість підйомів.

Ліміт часу **1** секунда

Ліміт використання пам'яті **128** МіВ

Вхідні дані #1

```
17  2
```

Вихідні дані #1

```
4
```

Вхідні дані #2

```
1  2
```

Вихідні дані #2

```
0
```

Вхідні дані #3

```
59049  3
```

Вихідні дані #3

```
9
```

Аналіз алгоритму

Промоделюємо процес руху котушки та підрахуємо кількість її підйомів.

Нехай l – поточна висота котушки. Вона після опускання підніматиметься вгору, тільки якщо рівень її підйому буде більше 1.
Якщо черговий підйом відбудеться на висоту, не більшу 1, то його вважати не будемо (котушка зупинилася).

Код програми

```
l,k = map(int, input().split())
s=0
while l>1:
    if l>k:
        s=s+1
    l=l/k
print(s)
```

Реалізація алгоритму

Читаємо вхідні данiКількість підйомів котушки підраховуємо у змінній s.

Нехай котушка знаходиться на висоті l. Тоді після опускання вона підніметься вгору, якщо висота її підйому буде більше 1 (інакше вважаємо, що котушка зупинилася). Це можливо за виконання нерівності l / k > 1

Виводимо відповідь.

Код програми

```
l,k = map(int, input().split())
s=0
while l>1:
    if l>k:
        s=s+1
    l=l/k
print(s)
```

Задача (B) Рівні дільники

Натуральне число **m** називається рівним дільником числа **n**, якщо частка і остача від ділення **n** на **m** рівні. За заданим натуральним числом **n** знайти кількість його рівних дільників.

Вхідні дані

Натуральне число **n** (**1 ≤ n ≤ 10^6**).

Вихідні дані

Надрукуйте одне шукане число.

Ліміт часу **1** секунда

Ліміт використання пам'яті **122.17** MiB

Вхідні дані #1	Вихідні дані #1
20	2

Аналіз і реалізація алгоритму

Оскільки n не велике, можна в циклі перебрати всі можливі значення i та перевірити виконання рівності n % i == n // i

Підраховуємо кількість k таких i

Код програми

```
n=int(input())
k=0
for i in range(1,n):
        if n % i == n // i:
            k=k+1
print(k)
```

Задача (C) Ледар

Наш Валера є класичним прикладом ледаря. На заняття він практично не ходить, і лише в кінці семестру появляється в університеті і здає ”хвости”. Його заповітна мрія: знайти такий день, коли можна буде здати відразу всі заборгованості. У нього є розклад роботи викладачів, з якого точно відомо, з якого і по який день місяця кожен викладач щоденно буде доступний. Допоможіть Валері написати програму, яка за розкладом буде визначати, чи зможе Валера здати всі заборгованості за один день чи ні.

Вхідні дані
Перший рядок вхідних даних містить кількість тестів. Кожен тест складається з числа N – кількості предметів, які потрібно здати Валері. Далі йде N рядків, кожен з яких складається з двох чисел A та B, які задають інтервал роботи чергового викладача.
Вихідні дані
Для кожного тесту програма повинна вивести у вихідний файл у окремому рядку "YES" якщо можливо зустріти всіх викладачів за один день, або "NO", якщо це неможливо.

Ліміт часу **1** секунда

Ліміт використання пам'яті **128** MiB

Вхідні дані #1

```
2
4
1 7
4 5
3 8
5 10
2
1 2
3 4
```

Вихідні дані #1

```
YES
NO
```

Аналіз і реалізація алгоритму

Нехай x – день, коли Валера зможе здати всі борги.

Нехай [ai, bi] (1 ≤ i ≤ n) – інтервали роботи викладачів.
Тоді очевидно, що має виконуватися
max(a1, …, an) ≤ x ≤ min(b1, …, bn)

Якщо max(a1, …, an) ≤ min(b1, …, bn), то необхідний день x існує, виводимо “YES”. Інакше виводимо "NO".

IDLE Shell 3.10.5

File Edit Shell Debug Options Window Help

```
Python 3.10.5
bit (AMD64)] c
Type "help", '
>>>
==============
2
4
1 7
4 5
3 8
5 10
YES
2
1 2
3 4
NO
>>>
```

c.py - D:/Python/Змагання цикли/c.py (3.10.5)

File Edit Format Run Options Window Help

```
tests=int(input())
while tests>0:
    n=int(input())
    min = 0
    max = 32
    for i in range(0,n):
        a,b = map(int, input().split())
        if a > min:
            min = a
        if b < max:
            max = b
    tests=tests-1
    if min<=max:
        print('YES')
    else:
        print('NO')
```

Код програми

C

```
tests=int(input())
while tests>0:
    n=int(input())
    min = 0
    max = 32
    for i in range(0,n):
        a,b = map(int, input().split())
        if a > min:
            min = a
        if b < max:
            max = b
    tests=tests-1
    if min<=max:
        print('YES')
    else:
        print('NO')
```

Задача (D) Поставка содової води

Тім дуже полюбляє содову воду, іннколи він нею ніяк не може напитися. Ще більш прикрим є той факт, що у нього постійно бракує грошей. Тому єдиним легальним способом їх отримання є продаж порожніх пляшок з-під соди. Іноді на додаток до його особисто випитих пляшок додаються ті, які Тім іноді знаходить на вулиці. Одного дня Тіма настільки замучила спрага, що він вирішив пити до тих пір поки міг собі це дозволити.

Вхідні дані

Три цілі невід'ємні числа e, f, c де e (e < 1000) - кількість порожніх пляшок, які є у Тіма на початку дня, f (f < 1000) - кількість порожніх пляшок, знайдених протягом дня, і c (1 < c < 2000) - кількість порожніх пляшок, необхідних для покупки нової пляшки.

Вихідні дані

Скільки пляшок содової води зможе випити Тім, коли його замучила спрага?

Ліміт часу **1** секунда

Ліміт використання пам'яті **128** МіБ

Вхідні дані #1

```
9 0 3
```

Вихідні дані #1

```
4
```

Вхідні дані #2

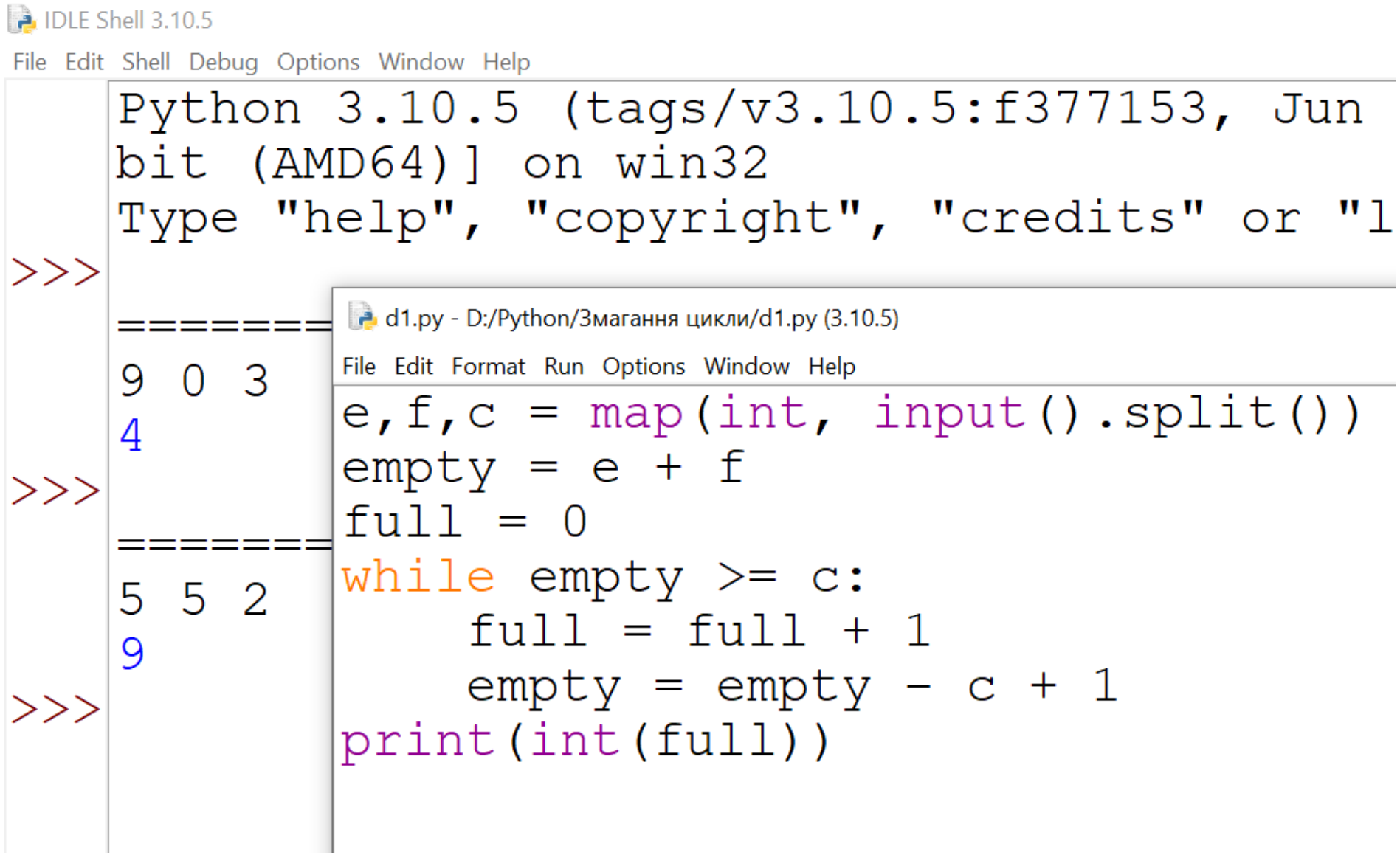
```
5 5 2
```

Вихідні дані #2

```
9
```

Аналіз і реалізація алгоритму

Розглянемо варіант, коли Тім здійснюватиме обмін по одній пляшці. Тобто якщо в нього є хоча б c пляшок, він іде та змінює їх на одну повну. Потім випиває її, після чого загальна кількість порожніх пляшок зменшується на $c - 1$.


IDLE Shell 3.10.5
File Edit Shell Debug Options Window Help
Python 3.10.5 (tags/v3.10.5:f377153, Jun
bit (AMD64)] on win32
Type "help", "copyright", "credits" or "l
>>>
========
9 0 3
4
>>>
========
5 5 2
9
>>>
d1.py - D:/Python/Змагання цикли/d1.py (3.10.5)
File Edit Format Run Options Window Help
e,f,c = map(int, input().split())
empty = e + f
full = 0
while empty >= c:
full = full + 1
empty = empty - c + 1
print(int(full))

Код програми

```
e,f,c = map(int, input().split())
empty = e + f
full = 0
while empty >= c:
    full = full + 1
    empty = empty - c + 1
print(int(full))
```

Аналіз і реалізація алгоритму

Завдання можна вирішити без використання циклів

Якщо спочатку кількість порожніх пляшок $e + f$ дорівнює 0, то відповідь 0. Якщо c – кількість порожніх пляшок, необхідних для купівлі нової пляшки, то вартість самої содової води, що знаходиться всередині однієї пляшки, становить $c – 1$ порожня пляшка. Спочатку у Тіма є $e+f$ порожніх пляшок, наприкінці у нього має залишитися не менше однієї порожньої пляшки. Тобто за $e+f – 1$ порожніх пляшок Тім може випити $(e+f – 1)/(c – 1)$ пляшок содової води.

Код програми

```python
e,f,c = map(int, input().split())
if f + e == 0:
    res = 0
else:
    res = (e + f - 1) / (c - 1)
print(int(res))
```

Задача (E) Відношення добутку до суми

Обчислити відношення добутку цифр натурального числа до їх суми.

Вхідні дані

Натуральне число **n**, що не перевищує $\mathbf{2 \cdot 10^9}$.

Вихідні дані

Вивести відношення добутку цифр числа **n** до їх суми з **3** десятковими цифрами.

Ліміт часу **1** секунда

Ліміт використання пам'яті **128** MiB

Вхідні дані #1

```
36
```

Вихідні дані #1

```
2.000
```

Вхідні дані #2

```
199999999
```

Вихідні дані #2

```
589681.110
```

Код програми

```
n=int(input())
s=0
d=1
while n > 0:
    s = s + n%10
    d = d * (n%10)
    n = n // 10
r=d/s
print("%.3f" % r)
```

Ліміт часу **1** секунда

Ліміт використання пам'яті **128** MiB

Задача (F) Дуже просто!!!

За заданими числами n та a обчислити значення суми: $\sum_{i=1}^{n} i \cdot a^i$

Вхідні дані
Два натуральні числа n та a.

Вихідні дані
Значення суми. Відомо, що вона не більша за 1018.

Вхідні дані #1

```
3 3
```

Вихідні дані #1

```
102
```

Вхідні дані #2

```
4 4
```

Вихідні дані #2

```
1252
```

Аналіз і реалізація алгоритму

Слід обчислити вказану суму за допомогою циклу. Якщо a ≥ 2, то значення n не буде занадто великим, тому що за умовою завдання сума не перевищує 1018.

Окремо слід обробити випадок a = 1, так як в цьому випадку при обчисленні циклом можна повчити Time Limit.

При a = 1 сума дорівнює

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

Код програми

```
n, a = map(int,input().split(' '))
s = 0
powa = 1
if a == 1:
  s = (1 + n) * n // 2
else:
  for i in range(1, n + 1):
    powa = powa * a;
    s = s + i * powa
print(s)
```

Задача (G) Кролики

Якось нарешті жителі планети Земля знайшли заселену планету, назвали її ТТВ, і відправили разом з кораблем туди одного кролика. Кролику сподобався клімат нової планети і через місяць він народив на світ ще одного кролика. Відомо, що кожен місяц кожен кролик, який був присутніс на планеті, відтворював на світ ще одного кролика. На планеті звідкілясь завіся монстр, який на початку місяця з'їдав ***k*** кроликів, як тільки їх ставало строго більше ***k***. У задачі необхідно визначити кількість кроликів, яка буде на планеті через ***n*** місяців після прибуття на неї космічного корабля з першим кроликом.

Вхідні дані

Перший рядок містить кількість місяців **n** (**0 ≤ n ≤ 100**), другий - число кроликів **k**(**0 ≤ k ≤ 10000**), яких з'їдав монстр.

Вихідні дані

Визначіть кількість кроликів, які будуть знаходитись на планеті **ТТВ** через ***n*** місяців після поселення на неї першого кролика. Відомо, що результат для довільного тесту завжди не перевищує $2*10^9$.

Ліміт часу **1** секунда

Ліміт використання пам'яті **64** MiB

Вхідні дані #1

```
0
10
```

Вихідні дані #1

```
1
```

Аналіз алгоритму

Достатньо промоделювати процес розмноження та поїдання кроликів.

Реалізація алгоритму

Читаємо вхідні дані.
Встановимо спочатку кількість кроликів res рівним 1.
Поки не пройде n місяців, моделюємо життя на планеті.
Поїдання монстром кроликів.
Розмноження кролів.
Виводимо кількість кроликів на планеті через n місяців

Код програми

```
n=int(input())
k=int(input())
res = 1
while n > 0:
    if res>k:
        res = res - k
    res = res * 2
    n=n-1
print(res)
```

Задача (P) Друга цифра числа

Знайти другу цифру цілого числа. Відлік починати з найвищого розряду.

Вхідні дані

Одне ціле **64** - розрядне число, що містить не менше двох цифр. Число може бути від'ємним.

Вихідні дані

Виведіть другу цифру заданого числа.

Ліміт часу **1** секунда

Ліміт використання пам'яті **128** MiB

Вхідні дані #1

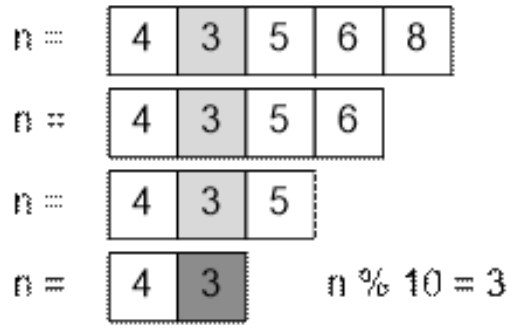
43568

Вихідні дані #1

3

Аналіз і реалізація алгоритму

Якщо вхідне число від’ємне, то обчислимо його модуль – від цього друга цифра не зміниться.
Далі ділимо число на 10, поки воно більше 99.
Остання цифра отриманого числа буде другою цифрою початкового числа.



Код програми

```
n = int(input())
if n < 0:
    n = -n
while n > 100:
    n = n // 10
res = n % 10
print(res)
```

Задача (I)
Число у зворотньому порядку

Записати ціле невід'ємне число **n** у зворотньому порядку.

Вхідні дані

Одне ціле невід'ємне 64-х розрядне число.

Вихідні дані

Запис числа у зворотньому порядку.

Ліміт часу **1** секунда

Ліміт використання пам'яті **128** MiB

Вхідні дані #1	Вихідні дані #1
1234	4321

Вхідні дані #2	Вихідні дані #2
100	001

Аналіз і реалізація алгоритму

Ділитимемо число на 10 поки не отримаємо 0.

На кожній ітерації виводимо останню цифру поточного числа.

Код програми

```
a = int(input())
if a<10:
    print(a)
else:
    while a > 0:
        print(a%10,end='')
        a=a//10
```

Задача (J) Кількість даних цифр в числі

Підрахувати.
Вхідні дані
У першому рядку записано одне ціле 32-розрядне число **n**.
У другому рядку записано одну цифру **a**.
Вихідні дані
Одне число - розв'язок задачі.

Ліміт часу **1** секунда

Ліміт використання пам'яті **128** MiB

Вхідні дані #1

```
25557
5
```

Вихідні дані #1

```
3
```

Вхідні дані #2

```
100
0
```

Вихідні дані #2

```
2
```

Аналіз і реалізація алгоритму

Ділитимемо число на 10 поки не отримаємо 0.

На кожній ітерації рахуємо кількість k цифр a в числі n

Виводимо результат k.

Код програми

```
n=abs(int(input()))
a=int(input())
k=0
while n > 0:
    if n%10==a:
        k=k+1
    n = n // 10
print(k)
```

Задача (К) Срібна медаль

Спортсмен Василь приймав участь у змаганнях з хокейболу і отримав в особистому заліку срібну медаль. Відомо, що учасники, які набрали однакову кількість очок, нагороджуються однаковими нагородами. Також відомо, що було розіграно золоті, срібні та бронзові медалі. У задачі не питають про правила хокейболу. Необхідно лише визначити, скільки очок набрав Василь.

Вхідні дані

У першому рядку задано кількість спортсменів **n** (**2 ≤ n ≤ 1000**), які приймали участь у змаганнях, у другому **n** цілих чисел - результати змагання.

Вихідні дані

Вивести одне число - результат Василя.

Ліміт часу **1** секунда

Ліміт використання пам'яті **128** MiB

Вхідні дані #1

```
5
4 3 3 1 2
```

Вихідні дані #1

```
3
```

Аналіз і реалізація алгоритму

Золотом нагороджуються учасники, які набрали максимальну кількість балів (max).

Сріблом нагороджуються учасники, які набрали найбільшу кількість очок, менше максимуму (max2).

У заданій послідовності потрібно знайти максимальне число, відмінне від найбільшого.

```
n=int(input())
a=list(map(int,input().split()))
max=a[0]
for i in range(n):
    if a[i]>max:
        max=a[i]
#print(max)
for i in range(n):
    if a[i]==max:
        a[i]=0
max2=a[0]
for i in range(n):
    if a[i]>max2:
        max2=a[i]
print(max2)
```

Задача (L) Казка про яблуко

Одного разу цар нагородив селянина яблуком зі свого садка. Прийшов селянин до садка і бачить: увесь сад загорожено N парканами, у кожному паркані лише одні ворота, і у кожних воротах стоїть сторож. Підійшов селянин до першого сторожа і показав царський указ, а сторож йому у відповідь: "_Йди візьми, але при виході віддаси мені половину тих яблук, які несеш, і ще одне_". Те ж саме йому сказали і другий, і третій сторож і т.д. Скільки яблук повинен узяти селянин, щоб після розрахунку зі сторожами у нього залишилось одне яблуко?

Вхідні дані

Єдине число N – кількість парканів у садку ($1 \le N \le 62$).

Вихідні дані

Єдине число K – кількість яблук, які повинен узяти селянин, щоб після розрахунку зі сторожами у нього залишилось одне яблуко.

Ліміт часу **1** секунда

Ліміт використання пам'яті **122.08** MiB

Вхідні дані #1	Вихідні дані #1
1	4

Вхідні дані #2	Вихідні дані #2
2	10

Аналіз і реалізація алгоритму

Нехай після проходу через чергову браму у селянина залишиться до яблук.
Отже, до проходу через ворота у нього мало бути (r + 1) * 2 яблук.
Повторюючи цю процедуру n разів, ми знайдемо початкову кількість яблук селянина.

Код програми

```
n = int(input())
r = 1
for i in range(n):
    r = (r + 1) * 2
print (r)
```

Задача (M) Відмінник Іванко

Ваня відмінник. Він радіє кожній п'ятірці, яку побачить у числі. Кожен ранок він їде на автобусі і рахує кількість п'ятірок у квитку, який йому попався. За давною прикметою (яка діє ще з 2-го класу), він знає, що за день отримає стільки п'ятірок, скільки їх у квитку. За номером сьогоднішнього квитка Іванка визначіть, скільки п'ятірок він отримає у цей день.

Вхідні дані
Номер квитка Іванка n ($0 \le n \le 9999$).

Вихідні дані
Виведіть кількість п'ятірок, яку отримає Іванко.

Ліміт часу **1** секунда

Ліміт використання пам'яті **128** MiB

Вхідні дані #1

3533

Вихідні дані #1

1

Аналіз і реалізація алгоритму

Вхідне число поміщаємо в цілий тип int.
Перебираємо цифри числа та підраховуємо кількість п'ятірок у ньому.
Перебір робимо, послідовно виконуючи розподіл на 10.
На кожній ітерації чергова цифра буде доступна як залишок від розподілу поточного числа на 10.

Код програми

```
n=int(input())
c = 0
while n > 0:
    if n%10==5:
        c = c + 1
    n = n // 10
print(c)
```

ЗМАГАННЯ

«ЦИКЛИ»

https://www.eolymp.com/uk/contests/6480