Submission

ID	DATE	PROBLEM	STATUS	CPU	LANG
	TEST CASES				
4790400	12:04:03	Trik	✓ **Accepted**	0.02 s	Python 3
	✓✓✓✓✓✓✓				

Submission contains 1 file: download zip archive

FILENAME	FILESIZE	SHA-1 SUM	
trik_260621270.py	288 bytes	04aabf091d3de1f159f97df191cd362dc7ba596e	download

Edit and resubmit this submission.

trik_260621270.py

```python
cupOrder = input()

cups = [1,0,0]

# switches order based on cupOrder.

for order in cupOrder:
  if order == 'A': cups[0], cups[1] = cups[1], cups[0]
  if order == 'B': cups[1], cups[2] = cups[2], cups[1]
  if order == 'C': cups[0], cups[2] = cups[2], cups[0]

print((cups.index(1) + 1))
```

Submission

ID	DATE	PROBLEM	STATUS	CPU	LANG
	TEST CASES				
4790407	12:05:17	The Easiest Problem Is This One	✓ **Accepted**	0.05 s	Python 3
	✓ ✓				

Submission contains 1 file: download zip archive

FILENAME	FILESIZE	SHA-1 SUM	
easiest_260621270.py	433 bytes	d832a0d9785eeef08dfd376c530211a32df8b641	download

Edit and resubmit this submission.

easiest_260621270.py

```
num = int(input())

# getting sum of digits
# add the mod of ten gets you the last digit
# return

def sum_digits(num):
  sum = 0
  while num > 0:
    rem = (num % 10)
    num = num // 10
    sum = int(sum) + rem
  return sum

while num != 0:
  sum_num = sum_digits(num)
  sum_product = 0
  p = 10
  while sum_product != sum_num:
    p += 1
    product = num * p
    sum_product = sum_digits(product)
  print(p)
  num = int(input())
```

Submission

ID	DATE	PROBLEM	STATUS	CPU	LANG
	TEST CASES				
4790427	12:08:04	I Can Guess the Data Structure!	✓ **Accepted**	0.15 s	Python 3
	✓✓				

Submission contains 1 file: download zip archive

FILENAME	FILESIZE	SHA-1 SUM	
datastructure_260621270.py	2798 bytes	1af1f79e15ea6d0c25488336babf1f24b02b9727	download

Edit and resubmit this submission.

datastructure_260621270.py

```
from sys import stdin

queue = []
stack = []
pri_queue = []

queue_status = True
stack_status = True
pri_queue_status = True

# checks to see if list is empty
# and if the element matches

def removeTup(line, queue_status, stack_status, pri_queue_status):
  tup = line.split()
  if(queue_status == True):
    if(len(queue) == 0 or queue[0] != str(tup[1])):
      queue_status = False
    else:
      queue.pop(0)
  if(stack_status == True):
    if(len(stack) == 0 or stack[len(stack)-1] != str(tup[1])):
      stack_status = False
    else:
      stack.pop()
  if(pri_queue_status == True):
    if(len(pri_queue) == 0 or pri_queue[len(pri_queue)-1] != int(tup[1])):
      pri_queue_status = False
    else:
```

```python
      pri_queue.pop()
  return queue_status, stack_status, pri_queue_status

# if input is an integer, it will be assigned to num
# otherwise, it is a line after the integer
# and it will be parsed
# originally started with tuples, but 2 digit numbers made it difficult

for line in stdin:
  try:
    num = int(line)
    queue_status = True
    stack_status = True
    pri_queue_status = True
  except ValueError:
    if (num == 1):
      tup = line.split()
      if(tup[0] == '1'):
        if(queue_status == True and stack_status == False and pri_queue_status == False):
          print("queue")
        elif(queue_status == False and stack_status == True and pri_queue_status == False):
          print("stack")
        elif(queue_status == False and stack_status == False and pri_queue_status == True):
          print("priority queue")
        elif(queue_status == False and stack_status == False and pri_queue_status == False):
          print("impossible")
        else:
          print("not sure")
      if(tup[0] == '2'):
        queue_status, stack_status, pri_queue_status = removeTup(line, queue_status, stack_status, pri_queue_status)
        if(queue_status == True and stack_status == False and pri_queue_status == False):
          print("queue")
        elif(queue_status == False and stack_status == True and pri_queue_status == False):
          print("stack")
        elif(queue_status == False and stack_status == False and pri_queue_status == True):
          print("priority queue")
        elif(queue_status == False and stack_status == False and pri_queue_status == False):
          print("impossible")
        else:
          print("not sure")
      queue = []
      stack = []
      pri_queue = []
      continue
    tup = line.split()
    if(tup[0] == '1'):
      queue.append(tup[1])
      stack.append(tup[1])
      pri_queue.append(int(tup[1]))
      pri_queue.sort(reverse=False)
    elif(tup[0] == '2'):
      tup = line.split()
      queue_status, stack_status, pri_queue_status = removeTup(line, queue_status, stack_status, pri_queue_status)
    num -= 1
```