Lottery @Csome

题目要求我们需要中奖彩票赚够1个亿，每个人初始金钱是1百万，只有完全中奖才能瓜分奖池，每期奖池是从每期购买的金钱总数累加的。

思路研究

```
def getWinningNumber():
    return tuple(map(lambda x: int(x, 16), hex(random.getrandbits(32))
[2:].rjust(8, '0')))
```

发现每期彩票来源是random.getrandbits(32)，想到python的randomCrack，

https://github.com/tna0y/Python-random-module-cracker

编写多线程脚本exp

```
# -*- coding: UTF-8 -*-
from pwn import *
import threading
import time
import randcrack
import sys

rc = {}
rc_lock = threading.Lock()


def f(t):
    # print(''.join(map(lambda x: f"{x:x}", t)))
    return int(''.join(map(lambda x: f"{x:x}", t)), 16)


def getWinningNumber(xt):
    return tuple(map(lambda x: int(x, 16), hex(xt)[2:].rjust(8, '0')))


class person:
    def __init__(self, personName, host, port):
        self.money = 0
        self.name = personName
        self.io = remote(host, port)
        self.io.sendlineafter('Your Name> ', self.name)
        self.hp = (host, port)

    def updateMoney(self):
        self.io.recvuntil("Your Money: ")
        self.money = float(self.io.recvuntil('$', drop=True).decode())

    def buyTicket(self, ticket, money):
        self.io.sendlineafter("> ", "1")
```

```
        self.io.sendlineafter('Your Ticket(Space interval)> ', ' '.join(map(str,
ticket)))
        self.io.sendlineafter('Your Spend> ', str(money))

    def waitForResult(self, save=1, Max_wait=300):
        while True and Max_wait:
            Max_wait -= 1
            tmp = self.io.recvuntil('\n').decode()
            if 'win' in tmp:
                break
            if "Error" in tmp:
                return False
            log.debug(self.name + " " + tmp)
        if not Max_wait:
            global NotEnough
            NotEnough = False
            return True
        if not save:
            return True
        self.io.recvuntil("Period: ")
        period = int(self.io.recvuntil('\n', drop=True).decode())
        self.io.recvuntil('Winning Number: ')
        number = f(eval(self.io.recvuntil('\n', drop=True).decode()))
        rc_lock.acquire()
        if period not in rc:
            rc[period] = number
        rc_lock.release()
        return True

    def buyFlag(self):
        self.io.sendlineafter("> ", "3")
        log.success(self.name.encode() + b" " + self.io.recvuntil('\n'))
        global NotEnough
        NotEnough = False

    def remake(self):
        self.io.close()
        self.io = remote(*self.hp)
        self.io.sendlineafter('Your Name> ', self.name)

    def attack(self):
        self.io.sendlineafter("> ", "1")
        self.io.recvuntil("Period ")
        period = int(self.io.recvuntil(' Award Pool: ', drop=True).decode())
        awardPool = int(float(self.io.recvuntil('$', drop=True).decode()))

        log.success(f"period:{period}")
        self.io.sendlineafter('Your Ticket(Space interval)> ', ' '.join(map(str,
winNumber[period])))
        self.io.sendlineafter('Your Spend> ', str(min(awardPool,
int(self.money))))


stat_time = time.time()
# host = "121.4.128.112"
```

```python
host = "0.0.0.0"
# port = 10023
port = 1234

NotEnough = True
flagPrice = 10 ** 8
MAX_PERSON_POOL = 20
LOG_LEVEL = "info"

th = []


def attackThread(attack=0, m=1, name="tmp", logLevel=LOG_LEVEL):
    context.log_level = logLevel
    pt = person(name, host, port)
    log.success(name)
    while NotEnough:
        pt.updateMoney()
        if attack and pt.money >= flagPrice:
            pt.buyFlag()
            break
        if attack:
            pt.attack()
            log.success("money" + str(pt.money))
        else:
            pt.buyTicket((1, 1, 1, 1, 1, 1, 1, 1), m)
        if not pt.waitForResult(not attack):
            pt.remake()
    pt.io.close()


for i in range(MAX_PERSON_POOL):
    t = threading.Thread(target=attackThread, args=(0, 1, f"Csome{i}"))
    t.start()
    th.append(t)

while len(rc) < 624:
    time.sleep(0.2)
    log.success(str(len(rc)))
NotEnough = False

for t in th:
    while t.is_alive():
        t.join(timeout=1)
        p1 = person(name, host, port)
        p1.buyTicket((1,1,1,1,1,1,1,1), 1)
        p1.io.close()

cr = randcrack.RandCrack()
print(len(rc))
rct = sorted(rc.items(), key=lambda x: x[0])

for i in range(len(rct) - 1):
    assert rct[i][0] + 1 == rct[i + 1][0]
```

```
tmp = []
for i, (idx, r) in enumerate(rct):
    if i < 624:
        cr.submit(r)
    else:
        tmp.append((idx, r))
for i, r in tmp:
    assert cr.predict_getrandbits(32) == r

winNumber = []
for i in range(1000):
    if (i + 1) % 100 == 0:
        print(i)
    winNumber.append((rct[-1][0] + i + 1,
getWinningNumber(cr.predict_getrandbits(32))))

winNumber = dict(winNumber)

NotEnough = True
threading.Thread(target=attackThread, args=(1, 100, "CsomeLyue",
'debug')).start()
for i in range(MAX_PERSON_POOL-1):
    threading.Thread(target=attackThread, args=(0, 10 ** 5, f"Lyue{i}")).start()


while NotEnough:
    time.sleep(0.2)
print(f"Spent Time: {time.time() - stat_time}")
```

即可得到flag