Problem 4

Kangaroo Checks

4 Points

Program Name: kangaroo.cpp   Input File: kangaroo.dat

At the end of each day, all banks must reconcile each account in its books. Reconciliation of an account is done by adding all the deposits to the starting balance and then subtracting all the debits to the account. Since the bank does not record the time that deposits and debits are made, the bank will give the benefit of the doubt to its customers so long as the ending balance for the day is not negative.

You are to write a program that reads the starting balances for up to 20 accounts followed by up to 200 transactions. Your program is supposed to determine which, if any, accounts end in negative balances (<0) at the end of the day. Any such account must be reported for further investigation. (The low number of accounts and transactions is because this is just a test of your program.)

Input

Input to your program will start with a single line containing the number of accounts at the bank (1<=A<=20). The following A lines each contain exactly 2 integers. The first integer, representing the account number, starts in column 1 and is exactly 6 digits in length with leading zeroes as necessary. The second integer is the starting balance of the account (in cents). The balance and all transactions are stored and processed as integers (in cents) because mathematical operations on floating point types are substantially more expensive than mathematical operations on the integer type. Starting balances may be positive or negative. Positive balances do not have a sign, but negative starting balances have a leading '-' sign.

Following the list of accounts and their starting balances are up to 200 transactions. Transaction lines start with a single character action field in column 1 ('C' for credit and 'D' for debit) followed by a blank in column 2 followed by the six-digit account number (with leading zeroes) followed by a blank in column 9 followed by the transaction amount (in cents) Your program should continue reading transactions until it reaches the end-of-file. You can be assured that there are no errors in the input file nor any extraneous spaces or blank lines.

Output

After reading all transactions, your program should print a list of accounts that end the day with negative balances. Each entry in the list should be the account number in columns 1-6 (with leading zeroes as necessary) followed by a single blank in column 7 followed by the balance of the account starting in column 8. Balances are shown in dollar format with parentheses around them to indicate a negative balance. For example, a balance of –42748 cents would be shown as ($427.48) because it is negative and has a decimal point because the balance is expressed in dollars. Your program should print both the parentheses and the dollar sign. The accounts should be ordered in the same order that they appeared in the input file relative to one another.

Example: Input File

```
12
322525 -803
353839 23923
484595 409439
002305 82385
103592 -9239
268358 190909
352920 58285
235235 2948234
000102 28523
253093 -123
111111 100000
325001 32952
C 268358 10293
D 322525 35252
D 353839 45823
D 111111 23253
D 353839 23455
D 268358 5000
D 111111 253998
D 322525 5000
D 268358 19382
D 322525 19203
D 352920 85492
C 268358 1092
D 268358 28345
C 322525 92385
C 111111 23902
D 353839 32523
C 353839 123952
D 111111 23528
C 103592 29395
D 353839 90934
D 322525 23852
C 353839 2535
C 268358 17398
C 322525 2359
D 353839 83499
```

Output to screen

```
353839 ($1258.24)
352920 ($272.07)
253093 ($1.23)
111111 ($1768.77)
```