Folding
Input File: FoldingIn.txt

Being tired of trying to jam as may items of clothing into her suitcase as possible. Maggie has purchased a newly invented digital suitcase to take with her on vacation. The suitcase accepts the clothing items in the form of a string of ASCII characters, `S`, folds them into a single signed integer `DS`, and then the integer is placed into the traveler's carry-on bag.

The folding process divides the ASCII characters of the string `S` into groupings of four sequential characters beginning with the first four characters in the string. Then these groupings are added to the integer `DS` one grouping at a time, ignoring overflow. If the last grouping does not contain four characters, the grouping is right justified and padded with zeros on the left to complete the 32 bit grouping. Spaces are considered ASCII characters.

For example, if Maggie was taking the following clothing items on vacation: *3 Pants*, the addition process would be as shown below. Note: the left most eight bits of the second grouping have been set to zero, because the grouping only contains three characters.

```
  3 Pa       00110011001000000101000001100001
+  nts     + 00000000011011100111010001110011
  xxxx       00110011100011101100010011010100 = DS = 864994516₁₀
```

If she were taking *2 Hats 3 Caps* the addition process would be as shown below, where zzzz and yyyy represent the intermediate subtotals:

```
  2 Ha
+ ts 3
  zzzz
+  Cap
  yyyy
+    s
  xxxx = DS
```

Inputs:
The first line of input contains the number of vacations to consider. This will be followed by one line of input per vacation that describes the clothing she will taking with her on that vacation.

Outputs:
There will be one line of output per vacation that contains the signed integer she places into her carry-on bag.

Sample Input

```
4
3 Pants
```

2 Hats 3 Caps
3 Pants 2 shorts 4 Coats
Caps

Sample Output

864994516
-959001993
-759357422
1130459251