CS 11 Exercise 05
1st Semester, AY2018-2019

University of the Philippines, Diliman

September 18, 2018

Instructions

- For this exercise, you have to write five different computer programs. A submission link is available in UVLe for each program.
- Make sure that your algorithm works given the sample input and output. You must also check if your algorithm can also handle input other than the ones given.
- Please remove any prompt messages (e.g. `Enter number: `) when getting input. Prompt messages will mess with your output, making your solution invalid.
- See the sample input and output to guide you on what and how your program must display output.
- Submit your solutions on or before Sunday, September 23 at 11:59pm.

1 Longest Word

Write a program that prints the longest word among a given a sequence of words.

1.1 Input

Input will start with the number of test cases t, followed be t test cases. Each test case t is a line consisting of one or more words, delimited by a space.

1.1.1 Sample Input

```
10
pumunta ako sa tindahan ni aling nena
para bumili ng suka
pagbayad ko aking nakita
isang dalagang nakadungaw sa bintana
natulala ako nalaglag puso ko
nalaglag din ang sukang hawak ko
napasigaw si aling nena
ako naman ay parang nakuryenteng pusa
ngunit natanggal ang hiya nang nakita ko
na nakatawa ang dalaga
```

1.2 Output

For each test case, output the longest word. If there are ties, output the first longest word from the left.

1.2.1 Sample Output

```
tindahan
bumili
pagbayad
nakadungaw
natulala
nalaglag
napasigaw
nakuryenteng
natanggal
nakatawa
```

2 Canonical Prime Factorization

Any natural number $N > 1$ can be written as a product of 1 or more powers of prime factors. Formally, for any natural number N, we can find $n \geq 1$ such that:

$$N = p_1^{a_1} p_2^{a_2} p_3^{a_3} \dots p_{n-1}^{a_{n-1}} p_n^{a_n}$$

Where p_i is a prime number and a_i is a positive integer. For example, the number 87660 can be written as

$$2^2 3^2 5^1 487^1$$

Likewise, the number 27000 can be written as

$$2^3 3^3 5^3$$

We call $p_1^{a_1} p_2^{a_2} p_3^{a_3} \dots p_{n-1}^{a_{n-1}} p_n^{a_n}$ the prime factorization of N. If the primes are ordered in increasing order, that is,

$$p_1 < p_2 < p_3 < \dots < p_{n-1} < p_n$$

then we call it a *canonical* prime factorization.

Write a program that prints canonical prime factorization of a given number N. Assume that $2 \leq N \leq 1000000$.

2.1 Input Format

Input will start with the number of test cases t, followed by t test cases. Each test case is a natural number N such that $2 \leq N \leq 1000000$.

2.1.1 Sample Input

```
18
128
5
306
4371
2264
7695
7696
10000
66672
44449
12352
1000000
48843
48841
15015
10
100
1000
```

2.2 Output Format

Output the canonical prime factorization of N. Print `'(p^a)'` for each prime power p^a in the factorization. The output for each test case must start in a new line.

2.2.1 Sample Output

```
(2^7)
(5^1)
(2^1)(3^2)(17^1)
(3^1)(31^1)(47^1)
(2^3)(283^1)
(3^4)(5^1)(19^1)
(2^4)(13^1)(37^1)
(2^4)(5^4)
(2^4)(3^2)(463^1)
(44449^1)
(2^6)(193^1)
(2^6)(5^6)
(3^6)(67^1)
(13^2)(17^2)
(3^1)(5^1)(7^1)(11^1)(13^1)
(2^1)(5^1)
(2^2)(5^2)
(2^3)(5^3)
```

3 Polynomial Addition

Given two polynomials, $P(x)$ of degree n and $Q(x)$ of degree m, the sum $P(x)+Q(x)$ is computed by adding the like terms. For example, for polynomials

$$\begin{aligned} P(x) &= x^7 + 4x^3 + 3x + 1 \\ Q(x) &= 3x^5 + 6x^3 + x + 20 \end{aligned}$$

the sum $P(x) + Q(x)$ is equal to

$$(1+0)x^7 + (0+3)x^5 + (4+6)x^3 + (3+1)x + (1+20)$$

which can then be simplified to

$$x^7 + 3x^5 + 10x^3 + 4x + 21$$

Write a program that computes the sum of two polynomials,

$$\begin{aligned} P(x) &= a_m x^m + a_{m-1}x^{m-1} + \ldots + a_2x^2 + a_1x^1 + a_0x^0 \\ Q(x) &= b_n x^n + b_{n-1}x^{n-1} + \ldots + b_2x^2 + b_1x^1 + b_0x^0 \end{aligned}$$

given their degrees m, n and the coefficients $a_0, a_1, \ldots a_{m-1}, a_m$ and $b_0, b_1, \ldots b_{n-1}, b_m$. Assume that $m, n \geq 0$ and n is not always equal to m.

3.1 Input Format

Input starts with the number of test cases t, followed by t test cases. Each test case is composed of two lines. The first line contains the degree and coefficients of P, and the second line contains the degree and coefficients of Q. On the line defining P, the first number is its degree m, followed by $m+1$ coefficients, starting at a_0 and ending at a_m. Likewise, on the line defining Q, the first number is its degree n, followed by $n+1$ coefficients, starting at b_0 and ending b_n. Assume that all inputs are integers, $m, n > 0$, $a_m \neq 0$, except when $m = 0$ and $b_n \neq 0$, except when $n = 0$.

3.1.1 Sample Input

```
8
7 1 3 0 4 0 0 0 1
5 20 1 0 6 0 3
4 -3 5 -20 0 3
4 1 4 9 6 2
2 1 2 1
5 1 2 3 0 0 -7
0 5
0 -7
2 3 5 3
2 3 6 -3
2 1 -2 1
2 1 -2 -1
3 -1 -2 -3 -4
0 0
0 5
0 -5
```

3.2 Output Format

For each test case, output $P(x)+Q(x)$, starting from the term with the highest exponent. For each nonzero term in the sum, output `'cx^i'`, where c is the sum of the coefficients for the term i. Note that some coefficents could be negative. If the first non-zero term is negative, leave it as is. For each succeeding non-zero term, output `' + '` or `' - '` before the term, depending on the result of adding the coefficients. If the term is positive, output `' + '`. If the term is negative, remove the sign and output `' - '`. In case the result of $P + Q$ is a zero polynomial, output `0x^0`.

3.2.1 Sample Output

```
1x^7 + 3x^5 + 10x^3 + 4x^1 + 21x^0
5x^4 + 6x^3 - 11x^2 + 9x^1 - 2x^0
-7x^5 + 4x^2 + 4x^1 + 2x^0
-2x^0
11x^1 + 6x^0
-4x^1 + 2x^0
-4x^3 - 3x^2 - 2x^1 - 1x^0
0x^0
```

4 Finite Continued Fractions

A finite continued fraction (written as $\langle a_0; a_1, a_2, \ldots, a_n \rangle$) is an expression of the form:

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ddots + \frac{1}{a_n}}}}$$

where n is a non-negative integer and $a_0, a_1, \ldots, a_n$ are integers such that $a_1, \ldots, a_n \geq 1$. We use the notation $\langle a_0; a_1, a_2, \ldots, a_n \rangle$ to denote a finite continued fraction.

Every rational number R has a finite continued fraction expansion. That is, for every real number R, there exists an integer $n \geq 0$ such that $R = \langle a_0; a_1, a_2, \ldots, a_n \rangle$. For example, the rational number $\frac{25}{9} \approx 2.7778$ has the following finite continued fraction expansion:

$$\langle 2; 1, 3, 2 \rangle = 2 + \cfrac{1}{1 + \frac{1}{3 + \frac{1}{2}}}$$

As another example, the rational number $\frac{57}{33} \approx 1.7273$ has the following finite continued fraction expansion:

$$\langle 1; 1, 2, 1, 2 \rangle = 1 + \cfrac{1}{1 + \frac{1}{2 + \frac{1}{1 + \frac{2}{4}}}}$$

Write a program that computes for the value of R, given the values of $a_0, a_1, \ldots, a_n$.

4.1 Input

Input starts with the number of test cases t, followed by t test cases. Each test case is a line of integers $a_0, a_1, \ldots, a_n$, where $a_1, \ldots, a_n \neq 0$.

4.1.1 Sample Input

```
10
5 4 3 2 1
2 1 3 2
1 1 2 1 2
1 2 2 1 4
0 2 4 6 8
7
0 1 2 3 4
7 14 21 28
0 3 3 3 3 3
0 9 9 9 9 9 9 9 9 9 9 9
```

4.2 Output

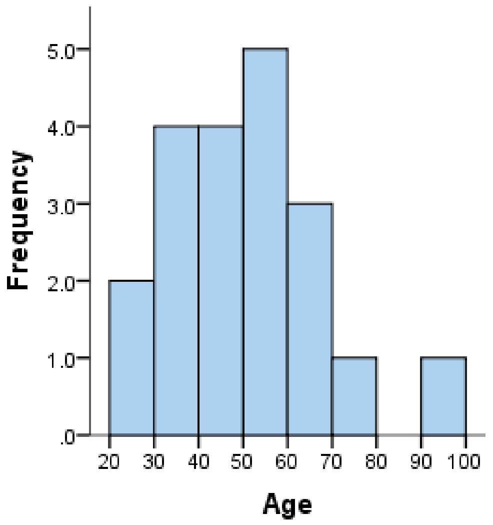
For each test case, output the number that corresponds to the finite continued fraction. Round your answer to 6 decimal places.

4.2.1 Sample Output

```
5.232558
2.777778
1.727273
1.424242
0.446389
7
0.697674
7.071187
0.302778
0.109772
```

5 Histogram

A histogram is a plot that lets you discover, and show, the underlying frequency distribution (shape) of a set of continuous data. This allows the inspection of the data for its underlying distribution (e.g., normal distribution), outliers, skewness, etc. An example of a histogram, and the raw data it was constructed from, is shown below:



Age: 36 25 38 46 55 68 72 55 36 38 67 45 20 48 91 46 52 61 58 55

To construct a histogram, you first need to split the data into intervals, called bins. In the example above, age has been split into bins, with each bin representing a 10-year period starting at 20 years. Each bin contains the number of occurrences of scores in the data set that are contained within that bin.

Write a program that creates a histogram of the numbers $a_0, a_1, \ldots, a_{n-1}, a_n$ Given bin size b.

5.1 Input

Input starts with the number of test cases t, followed by t test cases. Each test case consists of two lines. The first line contains the bin size b, the second line is the data. All inputs are integers, $b > 0$, and the data values are non-negative.

5.1.1 Sample Input

```
2
10
36 25 38 46 55 68 72 55 36 38 67 45 22 48 91 46 52 61 58 55
50
2 2 1 255 0 3 4 5 6 7 200 100 10 50 0
```

5.2 Output

For each bin in the histogram, output the bin interval, `‘l-h c’`, where `l-h` is the bin interval and ‘c’ is the number of items in the bin. For each bin, *l* and *h* denotes the lowest value and the highest value allowed in the histogram, respectively. The value of *l* in the first bin must be the lowest value in the data, and for all bins, and the start of succeeding bins must be *b* units away from the start of the previous bin. A blank line must follow the end of every test case.

5.3 Sample Output

```
22-31 2
32-41 4
42-51 4
52-61 6
62-71 2
72-81 1
82-91 1

0-49 11
50-99 1
100-149 1
150-199 0
200-249 1
250-299 1
```