

Przedmiotowy Konkurs Informatyczny LOGIA powołany przez Mazowieckiego Kuratora Oświaty

Zadanie Królik – LOGIA 25 (2024/25), etap 2

Treść zadania

Królik przygotowuje się do wysiewu marchwi. Pragnie, aby wszystkie marchewki wykiełkowały i nie chce zbyt długo czekać na pierwsze plony. Posiane zbyt płytko nie wejdą, posadzenie ich głębiej wydłuża czas oczekiwania na plony. Dlatego królik chce wyrównać głębokość dołków, w których sadi marchewki, w optymalny sposób. Jeśli choć jeden dołek jest za płytki, to pogłębia wszystkie za płytke dołki. W przeciwnym przypadku dosypuje ziemię w najpłytszych dołkach, tak aby miały minimalną głębokość, żeby marchewki wzeszły.

Pomóż królikowi i napisz program, który wczyta minimalną głębokość do posadzenia marchewki oraz głębokości dołków, a policzy łączną zmianę głębokości dołków.

Wejście

Pierwszy wiersz wejścia zawiera liczbę całkowitą g , która określa minimalną głębokość dołka, $1 \leq g \leq 1000$.

Drugi wiersz wejścia zawiera niepusty ciąg liczb całkowitych (maksymalnie 100 000) z zakresu od 1 do 2000, oddzielonych spacją, określających głębokości kolejnych dołków.

Wyjście

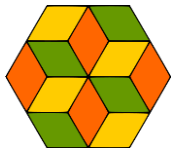
Jedna liczba całkowita określająca łączną zmianę głębokości dołków (suma pogłębień lub suma zmian głębokości dosypanych dołków).

Przykłady:

Wejście	Wyjście	Komentarz
20 8 30 15 40 20	17	Należy pogłębić pierwszy dołek o 12 i trzeci o 5.
12 30 40 50 40 30 60 70	36	Należy dosypać w pierwszym i piątym dołku po 18.
10 8 30 15 40 20	2	Należy pogłębić pierwszy dołek o 2.

Omówienie rozwiązania

Rozwiązanie zadania sprowadza się w zasadzie do znalezienia, czy w ciągu liczb całkowitych znajduje się liczba d mniejsza od minimalnej głębokości dołka g . Jeżeli taka liczba istnieje, to w dalszym obliczaniu sumujemy liczby $g-d$, czyli sumaryczne pogłębianie dołków. W przeciwnym przypadku, gdy wszystkie głębokości dołków są od d głębsze, to szukamy najmniejszej liczby i krotności jest wystąpienia.



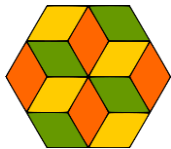
Przedmiotowy Konkurs Informatyczny LOGIA powołany przez Mazowieckiego Kuratora Oświaty

Zapis algorytmu w pseudokodzie może być następujący:

```
wczytaj g                                # optymalna głębokość
wczytaj tablicę G                        # głębokości dołków
minimum ← 2001                          # szukamy najpłytszego dołka
krotnosc ← 0
dla każdego elementu d w G powtarzaj:
  jeżeli d < minimum to
    minimum ← d
    krotnosc ← 1
  w przeciwnym przypadku
    jeżeli d = minimum to
      krotnosc ← krotnosc + 1
jeżeli minimum >= g to                    # dosypujemy
  wypisz krotnosc * (minimum - g)
w przeciwnym przypadku
  suma ← 0
  dla każdego elementu d w G powtarzaj:
    jeżeli d < g to
      suma ← suma + (g - d)                # suma pogłębianych dołków do g
  wypisz suma
```

Zauważmy, że tablicę **G** przeglądamy dwukrotnie. Pewnym wyzwaniem jest stworzenie algorytmu, który zwróci wynik po jednokrotnym przejrzaniu elementów w tablicy **G**. W tym celu można wykorzystać zmienną **krotnosc**, która ma przechowywać liczbę dołków głębszych, ale najbliższych głębokości **g**. Jak znajdzie się dołek do pogłębienia, nadamy jej wartość **-1** i od tego momentu jej wartość nie będzie się zmieniać, a będzie zwiększana wartość zmiennej **suma** – czyli o ile sumarycznie pogłębimy płytsze dołki. Zapis poprawionego algorytmu w pseudokodzie może być następujący:

```
wczytaj g
wczytaj tablicę G
minimum ← 2001
krotnosc ← 0
suma ← 0
dla każdego elementu d w G powtarzaj:
  jeżeli d >= g to
    jeżeli krotnosc >= 0 to
      jeżeli d < minimum to
        minimum ← d                    # dosypujemy
        krotnosc ← 1
      w przeciwnym przypadku
        jeżeli d = minimum to
          krotnosc ← krotnosc + 1      # kolejny dołek do dosypania
    w przeciwnym przypadku
      suma ← suma + g - d              # jednak pogłębiamy
      krotnosc ← -1
  jeżeli krotnosc = -1 to
    wypisz suma
  w przeciwnym przypadku
    wypisz krotnosc * (minimum - g)
```



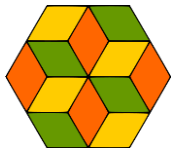
Przedmiotowy Konkurs Informatyczny LOGIA powołany przez Mazowieckiego Kuratora Oświaty

Rozwiązanie w języku Python

```
1 g = int(input())
2 suma, minimum, krotnosc = 0, 2001, 0
3 for d in map(int, input().split()):
4     if d >= g:
5         if krotnosc >= 0:
6             if d < minimum:
7                 minimum = d # dosypujemy
8                 krotnosc = 1
9             elif d == minimum:
10                krotnosc += 1 # kolejny dołek o tej głębokości dosypujemy
11     else:
12         suma += g - d # jednak pogłębiamy
13         krotnosc = -1
14 if krotnosc == -1:
15     print(suma)
16 else:
17     print(krotnosc * (minimum - g))
```

Testy

Grupa testów	Test	Wynik
I	50 50 55 60 65 70 75 80 85 90 95	0
	15 48 33 57 3 36 9 39 60 42 45 12 24 30 27 51 15 21 6 54 18	30
II	50 70 90 95 60 50 65 75 85 80 55	0
	15 78 81 51 90 30 21 72 45 33 57 24 48 39 87 42 18 66 69 84 75 27 60 54 63 36	3
III	500 754 530 805 860 792 797 760 542 826 883 643 827 669 720 878 560 713 625 900 626 771 586 863 866 530 648 837 867 667 684 536 530 871 875 631 854 534 567 628 530 551 548 674 835 871 543 530 530 899 615 694 863 530 530 719 530 891 805 530 578 746 682 608 839 688 645 530 704 835 692 530 530 808 791 530 836 530 643 701 629 686 855 845 673 819 893 554 530 632 746 632 530 774 550 759 530 608 530 782 530	600
	700 701 701 1083 779 701 701 701 701 701 701 773 991 1048 701 701 701 1060 790 701 701 881 701 877 701 701 701 701 701 701 746 785 701 1032 951 701 701 932 701 701 701 701 701 956 740 701 1000 701 1053 701 701 701 988 701 762 701 988 701 701 943 701 752 701 947 1007 701 1037 701 701 701 1018 701 701 701 701 878 701 1025 888 701 936 701 798 1080 1034 864 701 701 701 701 701 784 701 794 806 701 701 1056 701 701 701 701 701 701 701 701 701 881 935 909 701 928 701 890 770 701 1070 701 701 701 701 776 701 701 701 701 701 701 701 961 1045 701 949 912 976	120



Przedmiotowy Konkurs Informatyczny LOGIA powołany przez Mazowieckiego Kuratora Oświaty

	701 985 879 987 701 701 701 762 701 701 784 732 928 701 701 701 769 851 701 701 900 1027 701 1021 957 701 701 879 701 884 701 991 701 991 1088 875 701 1087 701 701 701 701 927 701 701 701 948 701 1019 701 920 701 701 770 701 701 701 857 701 701 701 912 701 883 701	
IV	100 101 827 818 131 1616 1887 101 604 565 101 1864 1303 202 1984 1890 374 101 613 453 1127 1974 1948 895 628 508 1051 673 1719 101 101 1017 966 101 710 723 101 514 537 1535 1311 101 1399 192 101 1400 623 101 774 1699 1197 101 101 101 246 162 1238 101 1312 101 772 1466 1312 860 1806 101 1506 741 960 101 1435 151 1414 910 101 1159 1985 1001 619 763 1814 1540 1096 1072 1657 1730 1572 784 101 1356 1732 839 1450 462 986 1589 465 1639 413 1005 1502	20
	100 99 99 58 99 99 68 76 99 99 27 99 56 37 40 99 99 61 20 99 34 46 99 9 99 99 99 16 96 99 88 99 99 56 99 99 99 62 99 99 5 12 98 99 67 99 99 93 99 99 20 92 99 99 99 5 35 99 21 86 99 99 99 57 99 99 99 35 99 43 99 99 99 99 62 49 19 99 91 99 2 99 6 99 72 99 99 41 1 99 99 99 99 99 99 57 92 99 99 99 58 99 99 99 99 99 13 99 99 99 70 99 99 36 99 99 71 28 73 24 99 99 99 73 99 99 99 99 99 36 99 14 80 99 99 99 99 99 11 99 99 99 99 99 99 99 30 99 38 99 99 40 99 32 64 63 99 22 45 86 99 3 58 99 99 39 99 99 99 89 5 41 99 99 99 99 99 84 94 84 80 99 99 99 99 99 28 99 93 42 99 99 99 99 72 99 99 85 99 99 99	4106
V	1000 100 000 losowo wygenerowanych liczb z zakresu od 1 do 2000	25067296
	1000 100 000 losowo wygenerowanych liczb z zakresu od 1 do 2000	25082980