Assalammualaikum

Mohammad Hanif Furqan Aufa Putra

5025221161

TUGAS 3 PBO C

Add All

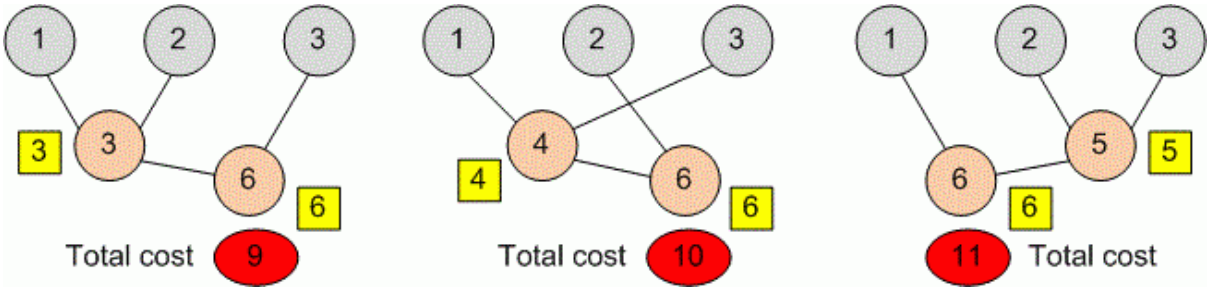
Time limit **2** seconds

Memory limit **128** MiB

The cost of adding two numbers equals to their sum. For example to add 1 and 10 costs 11. The cost of addition 1 and 2 is 3. We can add numbers in several ways:

- 1 + 2 = 3 (cost = 3), 3 + 3 = 6 (cost = 6), Total = 9
- 1 + 3 = 4 (cost = 4), 2 + 4 = 6 (cost = 6), Total = 10
- 2 + 3 = 5 (cost = 5), 1 + 5 = 6 (cost = 6), Total = 11

We hope you understood the task. You must add all numbers so that the total cost of summation will be the smallest.



Input data

First line contains positive integer n ($2 \le n \le 10^5$). Second line contains n nonnegative integers, each no more than 10^5.

Output data

Print the minimum total cost of summation.

Examples

Input example #1

```
3
1 2 3
```

Output example #1

```
9
```

Deskripsi masalah:

Soal ini melibatkan pencarian biaya total minimum untuk menambahkan sekumpulan angka dengan cara tertentu. Biaya penambahan dua angka sama dengan jumlah keduanya. Tujuannya adalah untuk meminimalkan total biaya penambahan dengan menemukan urutan penambahan sedemikian rupa sehingga biayanya diminimalkan.

Untuk menyelesaikan masalah ini, dapat menggunakan cara berikut:

1. Jika memiliki dua bilangan, A dan B, dapat menambahkannya dengan biaya A + B.

2. Jika memiliki tiga angka, A, B, dan C, dapat menjumlahkannya dengan biaya (A + B) + C atau A + (B + C). Untuk meminimalkan biaya, harus memilih dua angka dengan jumlah terkecil pada setiap langkah.

Dapat menggeneralisasi pendekatan ini ke kumpulan angka yang lebih besar:

1. Urutkan angka-angka dalam urutan menaik.

2. Ambil dua angka terkecil dan tambahkan bersama dengan biaya jumlah keduanya.

3. Ganti dua angka terkecil dengan jumlah keduanya.

4. 4. Ulangi langkah 2 dan 3 hingga Anda hanya memiliki satu angka yang tersisa, yaitu total biaya.

Sekarang, mari kita bahas bagaimana kode C++ yang disediakan menyelesaikan masalah ini.

Kode ini pada dasarnya mengimplementasikan pendekatan matematika yang telah dibahas sebelumnya. Kode ini secara efisien melacak nilai terbesar dan terkecil menggunakan antrian prioritas dan mendistribusikan ulang uang sampai semua penjaga memiliki jumlah yang sama.

Kelas `<vector>` digunakan untuk menyimpan angka-angka masukan, dan kelas `<queue>` digunakan untuk mengimplementasikan antrian prioritas. Antrian prioritas membantu dalam mengakses dengan cepat elemen-elemen maksimum dan minimum dalam himpunan angka, yang sangat penting untuk proses redistribusi.

Dibawah merupakan keseluruhan code

```cpp
#include <stdio.h>
#include <queue>
#include <functional>
using namespace std;
#define ll long long
#define gc getchar_unlocked

template <typename T> void Getnum(T &val){
    char ch; bool bo=0; val=0;
    for(ch=gc(); ch<'0'||'9'<ch; ch=gc()) if(ch=='-') bo=1;
    for(;'0'<=ch && ch<='9'; val=(val<<3)+(val<<1)+ch-48, ch=gc());
    if(bo) val = -val;
}

priority_queue<long, vector<long>, greater<long> > pq;

int main(){
    int n;
    ll num, res, i;
    Getnum<int>(n); // tadinya ll
    for(res = i = 0; i<n; i++)
        Getnum<ll>(num), pq.push(num);
    while(pq.size()>1){
        ll a = pq.top(); pq.pop();
        ll b = pq.top(); pq.pop();
        pq.push(a+b);
        res += a + b;
    }
    printf("%lld\n", res);
    return 0;
}
```

Dengan menjalan program tersebut diperoleh hasil seperti dibawah ini

Success #stdin #stdout 0.01s 5352KB

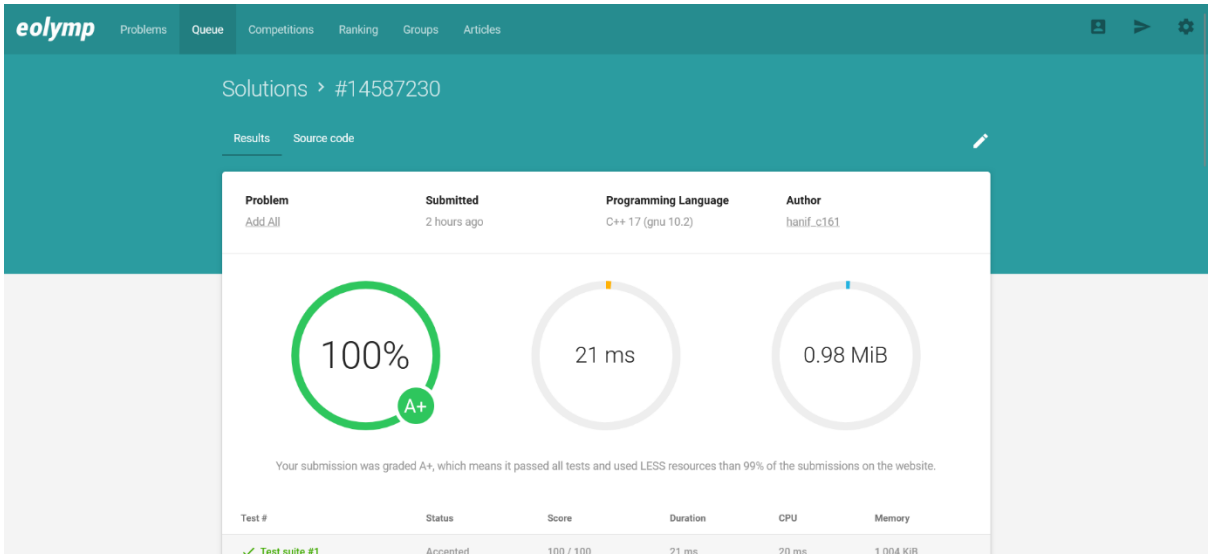
stdin

```
3
1 2 3
```

stdout

```
9
```

Dan dibawah bukti Verdict Accept



ACMCEG2C - Pick the candies

no tags

Pick the Candies

Many children went to a sweet shop. There were n candy varieties and each variety is kept in a separate bowl. The sweetness of each variety is written on the bowl. All the children wanted the candy with highest sweetness value. As there are only limited candies in each variety, the shop keeper makes a rule. According to the rule, the shopkeeper will show selectively chosen k varieties to every children. The children can pick any one of those varieties and move away. To make it easy for him, the shop keeper shows the

varieties 1,2,...,k to children1,

varieties 2,3,…,k+1 to children2,

varieties 3,4,…,k+2 to children3 and so on..

All the children are good at math. Find what variety each child will choose.

Input Specification:

The first line contains an integer t, the number of test cases. For each test case the input consists of two lines. The first line contains two integers n(number of candy varieties) and k. The next line contains n number of integers, the sweetness values of all the candy varieties.

Output Specification:

For each test case, print the candy varieties chosen by the children.

Input Constraints:

1<=t<=1000

1<=n<=10000

1<=k<=n

0<=Sweetness value<=10000

Sample Input:

3

5 3

1 2 3 4 5

4 2

7 1 6 8

9 5

7 14 3 0 2 2 2 2 2

Sample Output:

3 4 5

7 6 8

14 14 3 2 2

Hint: use deque

Disini kita memiliki sebuah toko permen dengan n jenis permen, dan setiap jenis memiliki nilai kemanisan tertentu. Beberapa anak datang ke toko tersebut, dan setiap anak ingin memilih permen dengan nilai kemanisan tertinggi. Namun, saya hanya dapat menunjukkan kepada mereka sejumlah (k) jenis permen dalam satu waktu. Saya harus mengikuti sebuah pola di mana untuk setiap anak, Saya menunjukkan sejumlah k jenis permen, dan anak tersebut dapat memilih dari pilihan tersebut. Tugas saya adalah menentukan jenis permen mana yang akan dipilih oleh setiap anak.

Saya akan menjelaskan berapa banyak permen yang bisa didapatkan setiap anak dan mengapa kasus uji coba ke-2 dan ke-3 berbeda dengan kasus uji coba ke-1 dan ke-2.

Pada soal, setiap anak hanya dapat memilih satu permen dari berbagai macam permen yang ditampilkan. Jadi, untuk setiap kasus uji, akan ada tepat n - k + 1 anak karena Anda memulai dengan menunjukkan permen pada anak pertama dalam rentang [1, k], lalu anak kedua mendapatkan permen dalam rentang [2, k + 1], dan seterusnya hingga anak terakhir mendapatkan permen dalam rentang [n - k + 1, n].

Sekarang, mari kita uraikan perbedaan antara kasus uji ke-1 dan ke-2 dibandingkan dengan kasus uji ke-3:

Kasus Uji Pertama:

- n = 5 (5 varietas permen)

- k = 3 (tampilkan 3 varietas sekaligus)

- Ada 5 - 3 + 1 = 3 anak dalam kasus uji ini.

- Anak-anak akan memilih permen dari rentang berikut:

  - Anak pertama: [1, 2, 3]

  - Anak kedua: [2, 3, 4]

  - Anak ketiga: [3, 4, 5]

Jadi, pada kasus uji pertama, ada 3 anak, dan setiap anak memilih satu permen dari rentang masing-masing.

Kasus Uji Coba ke-2:

- n = 4 (4 jenis permen)

- k = 2 (tampilkan 2 jenis permen dalam satu waktu)

- Ada 4 - 2 + 1 = 3 anak dalam kasus uji ini.

- Anak-anak akan memilih permen dari rentang berikut:

  - Anak pertama: [7, 1]

  - Anak kedua: [1, 6]

  - Anak ke-3 [6, 8]

Mirip dengan kasus uji pertama, pada kasus uji kedua, ada 3 anak, dan setiap anak memilih satu permen dari rentang masing-masing.

Kasus Uji ke-3:

- n = 9 (9 varietas permen)

- k = 5 (tampilkan 5 jenis permen dalam satu waktu)

- Ada 9 - 5 + 1 = 5 anak dalam kasus uji ini.

- Anak-anak akan memilih permen dari rentang berikut:

  - Anak pertama: [7, 14, 3, 0, 2]

  - Anak kedua: [14, 3, 0, 2, 2]

  - Anak ketiga: [3, 0, 2, 2, 2]

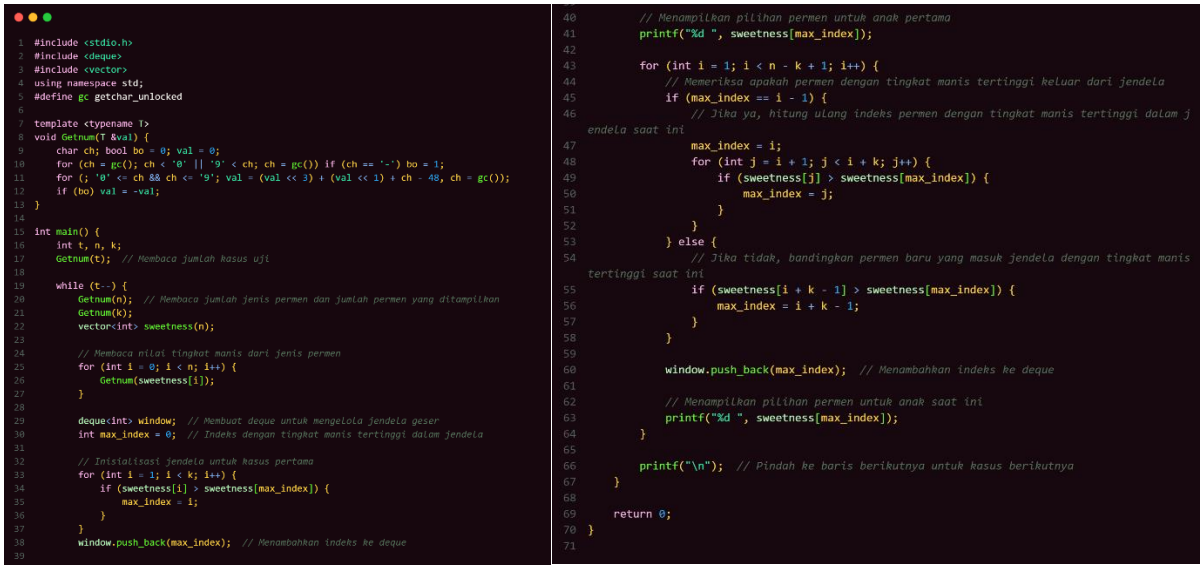
  - Anak ke-4: [0, 2, 2, 2, 2]

  - Anak ke-5: [2, 2, 2, 2, 2]

Pada kasus uji ke-3, terdapat 5 anak, dan setiap anak memilih satu permen dari rentang masing-masing.

Jadi, jumlah anak bervariasi berdasarkan nilai n dan k, dan setiap anak dalam setiap kasus uji memilih satu permen dari rentang yang telah ditentukan.

Dan dibawah ini merupakan keseluruhan code



```cpp
#include <stdio.h>
#include <deque>
#include <vector>
using namespace std;
#define gc getchar_unlocked

template <typename T>
void Getnum(T &val) {
    char ch; bool bo = 0; val = 0;
    for (ch = gc(); ch < '0' || '9' < ch; ch = gc()) if (ch == '-') bo = 1;
    for (; '0' <= ch && ch <= '9'; val = (val << 3) + (val << 1) + ch - 48, ch = gc());
    if (bo) val = -val;
}

int main() {
    int t, n, k;
    Getnum(t);  // Membaca jumlah kasus uji

    while (t--) {
        Getnum(n);  // Membaca jumlah jenis permen dan jumlah permen yang ditampilkan
        Getnum(k);
        vector<int> sweetness(n);

        // Membaca nilai tingkat manis dari jenis permen
        for (int i = 0; i < n; i++) {
            Getnum(sweetness[i]);
        }

        deque<int> window;  // Membuat deque untuk mengelola jendela geser
        int max_index = 0;  // Indeks dengan tingkat manis tertinggi dalam jendela

        // Inisialisasi jendela untuk kasus pertama
        for (int i = 1; i < k; i++) {
            if (sweetness[i] > sweetness[max_index]) {
                max_index = i;
            }
        }
        window.push_back(max_index);  // Menambahkan indeks ke deque

        // Menampilkan pilihan permen untuk anak pertama
        printf("%d ", sweetness[max_index]);

        for (int i = 1; i < n - k + 1; i++) {
            // Memeriksa apakah permen dengan tingkat manis tertinggi keluar dari jendela
            if (max_index == i - 1) {
                // Jika ya, hitung ulang indeks permen dengan tingkat manis tertinggi dalam jendela saat ini
                max_index = i;
                for (int j = i + 1; j < i + k; j++) {
                    if (sweetness[j] > sweetness[max_index]) {
                        max_index = j;
                    }
                }
            } else {
                // Jika tidak, bandingkan permen baru yang masuk jendela dengan tingkat manis tertinggi saat ini
                if (sweetness[i + k - 1] > sweetness[max_index]) {
                    max_index = i + k - 1;
                }
            }

            window.push_back(max_index);  // Menambahkan indeks ke deque

            // Menampilkan pilihan permen untuk anak saat ini
            printf("%d ", sweetness[max_index]);
        }

        printf("\n");  // Pindah ke baris berikutnya untuk kasus berikutnya
    }

    return 0;
}
```

Dengan menjalan program tersebut diperoleh hasil seperti dibawah ini

Success #stdin #stdout 0.01s 5312KB

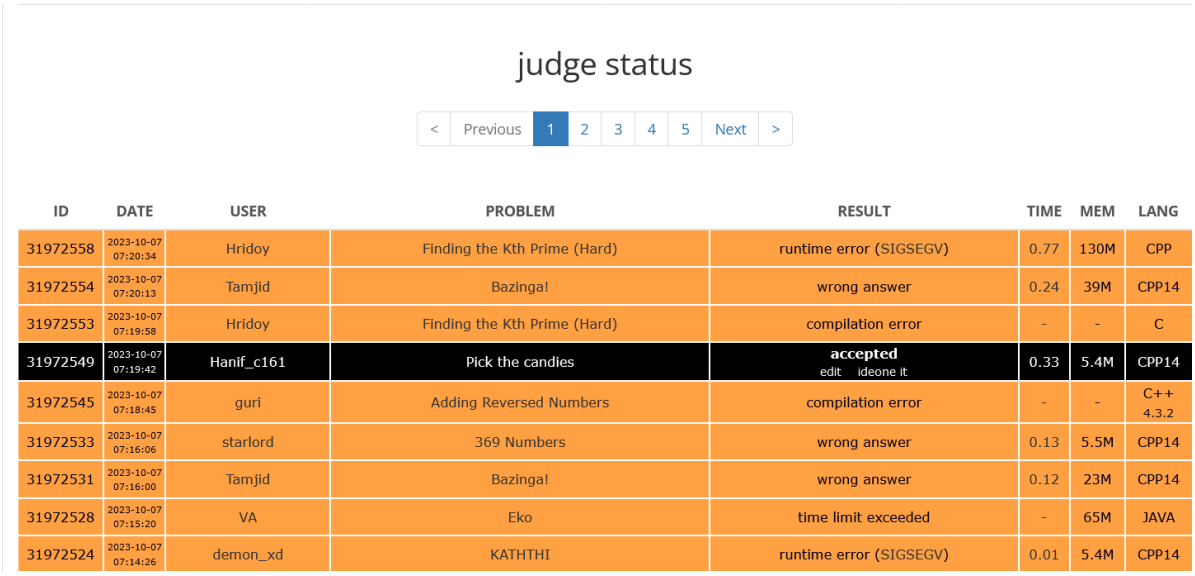
stdin

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

stdout

```
3 4 5
7 6 8
14 14 3 2 2
```

Dan dibawah bukti sudah Verdict Accept



judge status

< Previous 1 2 3 4 5 Next >

ID	DATE	USER	PROBLEM	RESULT	TIME	MEM	LANG
31972558	2023-10-07 07:20:34	Hridoy	Finding the Kth Prime (Hard)	runtime error (SIGSEGV)	0.77	130M	CPP
31972554	2023-10-07 07:20:13	Tamjid	Bazinga!	wrong answer	0.24	39M	CPP14
31972553	2023-10-07 07:19:58	Hridoy	Finding the Kth Prime (Hard)	compilation error	-	-	C
31972549	2023-10-07 07:19:42	Hanif_c161	Pick the candies	**accepted** edit ideone it	0.33	5.4M	CPP14
31972545	2023-10-07 07:18:45	guri	Adding Reversed Numbers	compilation error	-	-	C++ 4.3.2
31972533	2023-10-07 07:16:06	starlord	369 Numbers	wrong answer	0.13	5.5M	CPP14
31972531	2023-10-07 07:16:00	Tamjid	Bazinga!	wrong answer	0.12	23M	CPP14
31972528	2023-10-07 07:15:20	VA	Eko	time limit exceeded	-	65M	JAVA
31972524	2023-10-07 07:14:26	demon_xd	KATHTHI	runtime error (SIGSEGV)	0.01	5.4M	CPP14

Terima Kasih telah membaca, Waalaikumsalam wr wb.