Projeto e Análise de Algoritmos

Segundo Trabalho Prático (Primeira Versão)

Data Limite para Entrega: 18/01/2025

2º Semestre 2024 - DC-UFSCar

1 Introdução

No segundo Trabalho Prático (TP02) será solicitada a entrega de programas que solucionem os problemas apresentados nas próximas seções. Os programas entregues devem seguir os itens abaixo:

- Os programas devem ser feitos individualmente e plágio não será tolerado;
- Os programas devem ser entregues no beecrowd (https://www.beecrowd.com.br) com código em linguagem C, contendo um cabeçalho com as informações do estudante (nome, curso, RA);
- Cada estudante deve se cadastrar no beecrowd (https://www.beecrowd.com.br), acessar o beecrowd academic (https://www.beecrowd.com.br/judge/pt/disciplines) e se matricular na disciplina "PAA 2024s2 Turmas A, B e C" usando Chave CyWtic8 e ID 013672;
- Na linguagem C, compile seus programas usando o compilador GCC com flags -Wall -pedantic -O2 -Wno-unused-result, pois warnings podem impedir o código de funcionar no beecrowd ainda que funcionem no seu computador.
- Importante1: Para todos os estudantes de computação (engenharia ou ciência), gostaria que o trabalho fosse feito em linguagem C, como está especificado no enunciado.
- Importante2: Eu vou liberar uma lista para submissão em Python para que os estudantes de outros cursos possam usar esta linguagem, já que eles não veem introdução à programação em C.

Notem que os problemas que formam este trabalho prático fazem parte da lista de exercícios "Algoritmos Gulosos" na página da disciplina no beecrowd.

2 Limpando Filas

(1+1+1+1 = 4 pontos) Considere um sistema de atendimento que permite às pessoas entrar em uma fila para ser atendido de acordo com a ordem de chegada. Neste sistema, em que cada pessoa é identificada por um inteiro positivo, é possível entrar mais de uma vez na fila e, por algum motivo, as pessoas parecem adorar fazer isso. No entanto, não queremos que a mesma pessoa seja chamada inúmeras vezes. Assim, para implementar tal sistema é necessário um programa que limpe uma fila de tamanho N ($N \leq 10^7$) que contém duplicatas, preservando a ordem original e mantendo apenas a primeira vez em que uma pessoa entrou na fila. Além disso, é importante que a limpeza da fila seja feita bem rápido, pra não atrasar nos atendimentos dos clientes (o problema deve ser resolvido em tempo próximo a linear em N).

Entrada

A instância deve ser lida da entrada padrão. A primeira linha informa a quantidade N de entradas na fila. A segunda linha apresenta uma sequência com N números inteiros positivos, separados por espaços.

Saída

Deve ser impresso na saída padrão, em uma única linha e separados por espaços, a sequência correspondente à fila limpa de duplicatas. Esta linha deve ser terminada em '\n'.

Exemplos de entradas e saídas correspondentes

Entrada	Saída
6 1 3 5 7 3 1	1 3 5 7

Entrada	Saída
11 1 3 5 7 5 11 5 7 11 5 3	1 3 5 7 11

Desafio: Limpando Filas Twist

As pessoas realmente parecem gostar de entrar várias vezes na fila. Com o intuito de desincentivar este comportamento, houve uma mudança na regra. Agora a posição da pessoa após a limpeza será a da última vez em que ela entrou na fila, conforme os exemplos a seguir.

Entrada	Saída
6 1 3 5 7 3 1	5 7 3 1

Entrada	Saída
11 1 3 5 7 5 11 5 7 11 5 3	1 7 11 5 3

3 Black Friday

(1+1+1 = 3 pontos) Todo ano é a mesma coisa. A tal da black friday acaba se revelando uma black fraude. Mas não na loja do seu Florestan. Lá o desconto é pra valer!

A promoção de black friday na loja do seu Florestan foi no esquema compre 3, mas pague 2. Sempre que alguém passa no caixa com 3 ou mais produtos, leva o mais barato de graça. Como você não é nada bobo, decidiu passar várias vezes no caixa, cada uma delas com 3 produtos, para ganhar mais descontos. Como você não é nada bobo (mesmo!), quer aproveitar essa estratégia para conseguir o maior desconto possível.

Sua missão agora é criar um algoritmo que te indique qual é o maior desconto possível que você pode conseguir na black friday do seu Florestan.

Entrada

A instância deve ser lida da entrada padrão. A primeira linha indica o número N ($2 \leq N \leq 10^6$) de itens que você vai comprar na loja. A próxima linha contém os preços P_i ($1 \leq P_i \leq 10^7$), para $i = 1, \ldots, N$, de cada um dos itens da sua lista. Todos os valores de entrada são inteiros.

Saída

Ao final da execução, seu programa deve imprimir na saída padrão um único valor inteiro, indicando qual é o maior desconto que você conseguirá na promoção. Note que, esse maior desconto corresponde à soma dos valores de todos os itens pelos quais você não precisou pagar.

Exemplos de entradas e saídas correspondentes

Entrada	Saída
10 1 2 3 4 5 6 7 8 9 10	15

Entrada	Saída
9 2 1 1 2 2 3 3 3 1	6

Entrada	Saída
8 14 76 68 9 11 81 95 25	90

Desafio: Black Friday Twist

Seu Florestan endoidou e mudou a regra da black friday. Agora, cada vez que alguém passa no caixa com 3 ou mais produtos, leva o intermediário de graça, conforme os exemplos a seguir.

Entrada	Saída
10 1 2 3 4 5 6 7 8 9 10	21

Entrada	Saída
9 2 1 1 2 2 3 3 3 1	7

Entrada	Saída
8 14 76 68 9 11 81 95 25	149

4 Crush

(1,5+1,5 = 3 pontos) Poucas pessoas sabem qual foi a motivação de Edsger Dijkstra para inventar seu famoso algoritmo de caminhos mais curtos. Certo dia, ele estava estudando matemática quando sua crush mandou uma mensagem no Telegram dizendo "Oi, Ed! Estou em casa sem fazer nada e meus pais saíram. Mas eles voltam logo, vem rápido!". Para ter certeza que chegaria logo na casa da crush, ele precisou achar um caminho que fosse rápido até lá. Daí surgiu seu algoritmo.

É verdade. Eu estava lá. Eu era o celular do Dijkstra.

Como você já deve saber, Dijkstra, apesar de ser um dos melhores computeiros de todos os tempos, não é o melhor dos programadores. Ajude ele a fazer um programa que recebe um grafo ponderado e direcionado como entrada e que devolve a menor distância entre sua casa e a casa de sua crush. Assim, ele não perderá oportunidades únicas!

Entrada:

Cada entrada contém um único caso de teste. A primeira linha contém dois inteiros, o número de vértices V ($5 \leq$ V $\leq 10^6$) e o número de arcos E ($1 \leq$ E $\leq N(N-1)$) do grafo orientado. As próximas E linhas contêm as informações das arestas. Especificamente, cada linha tem três números inteiros, A ($0 \leq$ A $< V$), B ($0 \leq$ B $< V$) e W ($1 \leq$ W $\leq 10^4$), indicando que há uma aresta (direcionada) de A para B com peso W. A casa de Dijkstra se encontra no vértice 0 e a casa da crush se encontra no vértice $V-1$. É garantido que há um caminho entre esses dois vértices.

Saída:

Seu programa deve imprimir apenas uma linha, terminada em '\n', contendo um inteiro, indicando qual é a distância entre a casa do nosso amigo Edsger e seu (futuro) mozão.

Exemplos de entradas e saídas correspondentes

Entrada	Saída
8 13 0 3 7 1 0 3 2 1 3 2 4 6 2 5 6 3 5 2 4 7 5 5 1 6 5 6 3 5 7 6 6 2 9 6 7 1 7 6 3	13

Entrada	Saída
10 25 3 0 14 6 0 24 3 2 20 8 3 84 6 5 99 9 3 30 5 6 2 1 4 15 8 7 63 9 6 51 6 8 60 0 5 31 5 2 50 1 2 8 8 8 27 7 8 6 1 0 22 0 0 36 7 8 92 1 8 57 9 0 42 3 2 6 6 9 7 0 5 82 3 3 74	40