J.Koszelew

Zadanie 3: Ćwiczenia w stosowaniu technik programistycznych: technika zachłanna i programowanie dynamiczne

Rozwiązanie każdego z wariantów zadania polega na:

- **sprawdzeniu, czy problem opisany w treści zadania ma spełnioną własność optymalnej podstruktury i wyboru zachłannego,**
- **zaproponowaniu rozwiązania opartego na technice zachłannej, jeżeli nie zachodzi własność wyboru zachłannego a zachodzi własność optymalnej podstruktury,**
- **zaproponowaniu rozwiązania opartego na programowaniu dynamicznym, jeżeli spełnione są obydwie własności,**
- **określeniu pesymistycznej złożoności czasowej rozwiązania.**

Wariant A -„Kradzione nie tuczy”

Robotnicy budowlani pracujący na placu przy hurtowni piwa zauważyli, że magazyn hurtowni jest bardzo słabo pilnowany i bez problemu można z niego wynieść puszki piwa. Tylko jak? Wpadli na genialny pomysł. Zauważyli, że rury sanitarne, które mogli bez problemu przewozić przez bramę hurtowni mają średnicę niewiele większą od średnic puszek piwa. Postanowili upakować każdą z rur puszkami, a następnie wywieść tak „obciążone” rury z hurtowni.
Każda z puszek piwa miała znaną cenę, więc fachowcy dbali też o to, aby w rurach znalazły się najcenniejsze z nich. Dla niepoznaki każda rura miała nałożone obustronne zaślepki, aby strażnik nie zauważył jej wnętrza. Spryt fachowców nie znał granic, więc zadbali też o to, aby każda z rur była maksymalnie upakowana, czyli suma wysokości puszek w rurze była równa wysokości całej rury. Wysokość puszek, które mogły być włożone do rury oraz wysokość rur jest z założenia pewną całkowitą potęgą dwójki (2^i, gdzie $1 \le i \le 1000$).
Nie jest to moralnie poprawne, ale napisz program, który:

- po podaniu danych określających wysokości rur wyrażonych wykładnikami potęg dwójki oraz danym asortymencie puszek, określającym ich wysokości (również w postaci wykładników potęg dwójki) oraz wartości,
- sprawdzi, czy możliwe jest upakowanie wszystkich rur puszkami. Jeżeli sposobów upakowania będzie kilka, to należy wybrać taki, który ma największą wartość łączną puszek.

Wejście
W pierwszej linii pliku tekstowego KT_in_grupa_nazwisko.txt zapisana jest jedna liczba n - liczba puszek w magazynie n, która spełnia nierówność: $1 \le n \le 1000000$. W następnych n liniach znajdują się dwójki liczb całkowitych nieujemnych i oraz w oddzielone spacją: i - wysokość puszki, w - wartość puszki. Wiadomo, że wysokość jest potęgą dwójki, zatem i jest wykładnikiem tej potęgi. Wartość puszki w spełnia warunek $w \le 10000$. W kolejnej linii pliku znajduje się dodatnia liczba całkowita t określająca, ile różnych wysokości mają rury, którymi dysponują fachowcy. W następnych t liniach zapisane są pary dodatnich liczb całkowitych oddzielonych pojedynczym odstępem. Pierwsza z tych liczb l jest wysokością rury (właściwie

wykładnikiem potęgi dwójki), natomiast druga liczba d oznacza liczbę dostępnych rur tej wysokości. Zakładamy, że $t \leq 5000$, a $l \leq 1000$.

Wyjście
W pliku KT_out_grupa_nazwisko.txt należy wypisać napis NIE, jeżeli nie istnieje możliwość upakowania wszystkich rur puszkami albo liczba całkowita równa maksymalnej, łącznej wartości puszek z wszystkich rur, o ile można nimi szczelnie wypełnić wszystkie rury z podanego zestawu.

Przykład

KT_in_grupa_nazwisko.txt:
5
1 4
2 6
1 3
2 2
1 5
2
1 1
2 2

KT_out_grupa_nazwisko.txt:
18

Wariant B – „Problem kasjera w wersji z ograniczeniami"

Środkiem płatniczym w pewnym kraju są banknoty o nominałach $b_1, b_2, \ldots, b_n$. Kasjer w hipermarkecie ma zawsze nie lada stres gdy wydaje resztę – boi się, że zabraknie mu w pewnym momencie banknotów któregoś z nominałów. W celu ograniczenia poziomu stresu kasjer postanowił wydawać każdą resztę używając jak najmniejszej liczby banknotów.
Napisz program, który:

- wczyta opis zapasu banknotów, które posiada kasjer oraz kwotę do wypłacenia,
- obliczy minimalną łączną liczbę banknotów, za pomocą jakiej kasjer może wypłacić zadaną kwotę oraz znajdzie pewien sposób jej wypłacenia,
- wypisze wynik w pliku.

Wejście:
W pierwszym wierszu pliku PKO_in_grupa_nazwisko.txt znajduje się liczba nominałów n, $1 \leq n \leq 200$. Drugi wiersz zawiera n liczb całkowitych $b_1, b_2, \ldots, b_n$, $1 \leq b_1 \leq b_2 \leq \ldots \leq b_n \leq 20000$, pooddzielanych pojedynczymi odstępami. Trzeci wiersz zawiera n liczb całkowitych $c_1, c_2, \ldots, c_n$, $1 \leq c_i \leq 20000$, pooddzielanych pojedynczymi odstępami; c_i jest liczbą banknotów o nominale b_i znajdujących się w bankomacie. W ostatnim, czwartym wierszu wejścia znajduje się jedna liczba całkowita k — kwota, którą bankomat ma wypłacić, $1 \leq k \leq 20000$. Możesz założyć, że kwotę k można wypłacić za pomocą dostępnych banknotów.

Wyjście:
Pierwszy wiersz pliku tekstowego PKO_out_grupa_nazwisko.txt powinien zawierać jedną dodatnią liczbę całkowitą równą minimalnej łącznej liczbie banknotów, za pomocą których bankomat może wypłacić kwotę *k*. Drugi wiersz powinien zawierać *n* liczb całkowitych, oddzielonych pojedynczymi odstępami i oznaczających liczby sztuk poszczególnych banknotów użytych do wypłacenia kwoty *k*. W przypadku, gdy istnieje więcej niż jedno rozwiązanie, program powinien wypisać którekolwiek.

Przykład

PKO_in_nazwisko_grupa.txt:
3
2 3 5
2 2 1
10

PKO_out_nazwisko_grupa.txt:
3
1 1 1

Wariant C – „Studencka wycieczka”

Studenci II roku informatyki WI PB zaliczyli pomyślnie sesję zimową i wybierają się na wycieczkę po Europie. Co prawda lotem bliżej, ale na pewno nie taniej, więc jako środek transportu uwzględniają wyłącznie autokar. Spędzą w drodze wiele dni, zatem opłaty za noclegi stanowią poważną część kosztów podróży. Ze względu na bezpieczeństwo podróży każdy autokar jedzie tylko w ciągu dnia i nie może przejechać więcej niż 800 km. Noc na trasie (poza początkiem i końcem) studenci i kierowcy spędzają w hotelach. Oczywiście trzeba podróż zaplanować optymalnie, czyli tak, aby koszt noclegów w ciągu całej wycieczki był jak najmniejszy. Przedsiębiorstwo przewozowe znając potrzebę obniżenia kosztów wycieczki, postanowiło zbadać, czy opłaci się układanie planów podróży tak, aby suma opłat za noclegi była możliwie najniższa, nawet wtedy gdyby to miało przedłużyć podróż. W każdej ofercie jest podana odległość hotelu od początku trasy i cena jednostkowa noclegu. Studenci podróżują tylko w jedną stronę. Trasa nie ma rozgałęzień. Przez każdy punkt trasy autokar przejeżdża tylko raz. Nie ma dwóch hoteli w jednym punkcie, więc każdy hotel może być zidentyfikowany przez jego odległość od początku trasy. Nie planuje się noclegu ani na początku ani na końcu trasy. Liczba osób w autokarze nie zmienia się i wszyscy razem z kierowcą płacą taką samą cenę jednostkową za nocleg w tym samym hotelu. Pojemność hoteli jest na tyle duża, że wszyscy mogą w nim zanocować. Na każdym odcinku trasy o długości 800 km jest przynajmniej jeden hotel.

Pomóż przedsiębiorstwu, pisząc program, który:

- po wczytaniu długości trasy, liczby hoteli oraz oferty hoteli (cena jednego noclegu za jedną osobę)

- znajduje plan najtańszej podróży, tzn. takiej, której suma opłat za hotele była najmniejsza, a jeśli takich rozwiązań jest wiele, wybiera jedno z najmniejszą liczbą noclegów.

Wejście:
W pliku wejściowym SW_in_grupa_nazwisko.txt zapisane są dwie liczby całkowite dodatnie d i h oznaczające odpowiednio długość trasy i liczbę hoteli na trasie. Zakładamy, że długość trasy jest liczbą całkowitą $d \leq 16000$, a liczba hoteli $h \leq 1000$. Zakładamy też, że podane pary są posortowane rosnąco względem wartości d.

Wyjście:
W pierwszej linii pliku wynikowego SW_out_nazwisko.txt zapisana jest liczba całkowita w oznaczająca liczbę wybranych hoteli. W kolejnych h liniach pliku znajdują się liczby całkowite dodatnie oznaczające odległość wybranych hoteli od początku trasy.

Przykład

SW_in_grupa_nazwisko.txt:
2000 7

100 54

120 70

400 17

700 38

1000 25

1200 18

1440 40

SW_out_grupa_nazwisko.txt:
2

400 1200

Wariant D - " Remonty, remonty …"

Wszystkie sale WI będą remontowane przez cały semestr zimowy. Wszystkie, oprócz jednej auli. W tej jednej auli będą się odbywały wszystkie wykłady. Oczywiście nie jest to możliwe, zwłaszcza, że wszyscy wykładowcy chcą mieć swoje wykłady tego samego dnia. Mało tego, wykładowcy, którzy chcą korzystać z sali, składają zamówienia określając czas rozpoczęcia i zakończenia wykładu. Biedny Pan Tomek, który ma się zająć jak zwykle rozkładem, ma ułożyć plan wykorzystania auli akceptując pewne wykłady i odrzucając inne, tak, aby czas wykorzystania sali był jak najdłuższy. Zakładamy, że w momencie zakończenia jednego wykładu może się rozpocząć następny wykład.

Pomóż Panu Tomkowi i napisz program, który:

- na podstawie życzeń wykładowców,
- wybiera takie wykłady, których łączny czas trwania będzie najdłuższy.

Wejście:
W pierwszej linii pliku wejściowego RR_in_grupa_nazwisko.txt zapisana jest liczba życzeń wykładowców n ($1 \leq n \leq 1000000$). W następnych n liniach pliku zapisane są pary liczb całkowitych dodatnich p i k oddzielone pojedyncza spacją oznaczające odpowiednio początek zajęć i koniec zajęć ($0 \leq p \leq k \leq 30000$).

Wyjście:
W pierwszej i jedynej linii pliku wynikowego RR_out_grupa_nazwisko.txt zapisana jest liczba całkowita nieujemna oznaczająca maksymalny czas łączny trwania wybranych wykładów

Przykład

RR_in_grupa_nazwisko.txt:
12

1 2

3 5

0 4

6 8

7 13

4 6

9 10

9 12

11 14

15 19

14 16

18 20

RR_out_grupa_nazwisko.txt:
16