バス通学 (Bus)

大学生の JOI 君はバスで通学している．JOI 君の自宅も JOI 君の通う大学も IOI 市内にある．IOI 市には N 個のバス停があり，1 から N までの番号が付いている．JOI 君の自宅の最寄りはバス停 1 で，大学の最寄りはバス停 N である．

IOI 市内を走るバスは M 本あり，それぞれのバスは 1 日に 1 回，定められた時刻に定められたバス停を出発し，定められた時刻に定められたバス停に到着する．日付をまたぐような運行を行うバスは存在しない．JOI 君はバスに途中で乗ったりバスから途中で降りたりすることはできない．

JOI 君は毎日，１本以上のバスを乗り継いで大学に通っている．JOI 君がバスを乗り換えるのにかかる時間は無視できるとする．すなわちある時刻にあるバス停を出発するバスに乗り換えるためには，そのバスの出発時刻またはそれ以前にそのバス停に到着していればよい．また，同じバス停を複数回利用してもよい．

そのような条件のもとで，JOI 君はいつ家を出れば授業に間に合うように大学に到着できるかを知りたい．ただし大学は 1 日の初めの授業が開始する時刻が日によって異なる．ある Q 日間について，その日の授業に間に合うためにはバス停 N にいつまでに到着すればよいかがわかっている．それぞれの日について，JOI 君は遅くともいつまでにバス停 1 に到着すれば授業に間に合うだろうか．

課題

バスの運行に関する情報が与えられる．さらにある Q 日間について，バス停 N にいつまでに到着すればよいかが与えられるので，それぞれについて，JOI 君が遅くともいつまでにバス停 1 に到着すればよいかを求めよ．

入力

標準入力から以下の入力を読み込め．

- １行目には，２つの整数 N, M が空白を区切りとして書かれており，IOI 市内には N 個のバス停があり M 本のバスが走っていることを表す．
- 続く M 行のうちの i 行目 $(1 \leqq i \leqq M)$ には，4 つの整数 A_i, B_i, X_i, Y_i $(1 \leqq A_i \leqq N, 1 \leqq B_i \leqq N, A_i \neq B_i)$ が空白を区切りとして書かれている．これは i 番目のバスが，バス停 A_i を時刻 X_i に出発し，バス停 B_i に時刻 Y_i に到着することを表す．ただし時刻は午前 0 時ちょうどからの経過時間をミリ秒単位で表したものである．
- 次の行には整数 Q が書かれている．これは，バス停 N にいつまでに到着すればよいかが与えられる日数が Q 日間であることを表す．
- 続く Q 行のうちの j 行目 $(1 \leqq j \leqq Q)$ には，整数 L_j が書かれている．これは j 番目の日にはバス停 N に時刻 L_j までに到着しなければならないことを示す．

出力

標準出力に Q 行出力せよ．j 行目 ($1 \leqq j \leqq Q$) に，j 番目の日に JOI 君が遅くともいつまでにバス停 1 に到着すればよいかを表す整数を出力せよ．ただし，授業に間に合うように大学に到着することが不可能なときは -1 を出力せよ．

制限

すべての入力データは以下の条件を満たす．

- $2 \leqq N \leqq 100\,000$.
- $1 \leqq M \leqq 300\,000$.
- $0 \leqq X_i < Y_i < 86\,400\,000$ $(= 24 \times 60 \times 60 \times 1000)$ $(1 \leqq i \leqq M)$.
- $1 \leqq Q \leqq 100\,000$.
- $0 \leqq L_j < 86\,400\,000$ $(= 24 \times 60 \times 60 \times 1000)$ $(1 \leqq j \leqq Q)$.

小課題

小課題 1 [20 点]

以下の条件を満たす．

- $N \leqq 2\,000$.
- $M \leqq 2\,000$.
- $Q = 1$.

小課題 2 [15 点]

以下の条件を満たす．

- $N \leqq 2\,000$.
- $M \leqq 2\,000$.

小課題 3 [15 点]

- $Q = 1$ を満たす．

小課題 4 [50 点]

追加の制限はない.

入出力例

入力例 1	出力例 1
5 6 1 2 10 25 1 2 12 30 2 5 26 50 1 5 5 20 1 4 30 40 4 5 50 70 4 10 30 60 100	-1 5 10 30

バス停 5 に時刻 10 までに到着することはできない.
時刻 30 までに到着するためには，時刻 5 に 4 番目のバスに乗ればよい.
時刻 60 までに到着するためには次のようにすればよい.

- 時刻 10 に 1 番目のバスに乗車.
- 時刻 25 にバス停 2 に到着し，1 ミリ秒待って 3 番目のバスに乗車.
- 時刻 50 にバス停 5 に到着.

時刻 100 までに到着するためには次のようにすればよい.

- 時刻 30 に 5 番目のバスに乗車.
- 時刻 40 にバス停 4 に到着し，10 ミリ秒待って 6 番目のバスに乗車.
- 時刻 70 にバス停 5 に到着.

入力例 2	出力例 2
3 8 1 2 1 5 1 3 0 1 1 3 2 8 2 3 2 3 2 3 3 4 2 3 4 5 2 3 5 6 2 3 6 7 6 3 4 5 6 7 8	0 0 0 1 1 2

たのしい家庭菜園 (Growing Vegetables is Fun)

家庭菜園が趣味の JOI 君は，毎年，自分の畑で IOI 草という植物を育てている．JOI 君の畑は東西方向に並んだ N 個の区画に分かれており，西側から順に 1 から N まで番号が付いている．IOI 草は全部で N 株あり，それぞれの区画に 1 株ずつ植えてある．区画 i に植えられている IOI 草は，春になると背丈 h_i まで伸び，その後は伸びない．

春になって様子を見に行った JOI 君は，IOI 草の配置が予定とは異なる配置になっていることに気づいた．IOI 草は日光を多く必要とする植物で，ある区画に植えられている IOI 草について，その区画より番号の小さい区画と番号の大きい区画の両方にその IOI 草より背丈の高い IOI 草があると，その IOI 草は夏になる前に枯れてしまう．すなわち，どの IOI 草も枯れないようにするためには，以下の条件が満たされなければならない．

- $2 \leqq i \leqq N-1$ を満たすどの整数 i についても，以下の 2 つの条件のうち少なくとも 1 つが成立する．
  - $1 \leqq j \leqq i-1$ を満たすすべての整数 j について，$h_j \leqq h_i$ を満たす．
  - $i+1 \leqq k \leqq N$ を満たすすべての整数 k について，$h_k \leqq h_i$ を満たす．

IOI 草は大変高価なため，どの IOI 草も枯れないように JOI 君は IOI 草の並べ替えを行うことにした．IOI 草は大変大きく，かつ繊細な植物なため，JOI 君は隣り合う 2 つの IOI 草を並べ替えることしかできない．つまり，1 回の操作で JOI 君は区画 i $(1 \leqq i \leqq N-1)$ を任意に 1 つ選び，区画 i の IOI 草と区画 $i+1$ の IOI 草を入れ替えることができる．夏が近づくにつれて枯れる可能性が高くなるので，すべての IOI 草が枯れないようにするために必要な操作回数の最小値が知りたい．

課題

JOI 君の畑の区画数と，それぞれの IOI 草の背丈の情報が与えられたとき，すべての IOI 草が枯れないように並べ替えるために必要な操作回数の最小値を求めるプログラムを作成せよ．

入力

標準入力から以下の入力を読み込め．

- 1 行目には整数 N が書かれている．N は JOI 君の畑の区画数を表す．
- 続く N 行には，IOI 草の背丈に関する情報が書かれている．これらの行のうち i 行目 $(1 \leqq i \leqq N)$ には整数 D_i が書かれている．D_i は区画 i に植えられている IOI 草の春になった時点での背丈を表す．

出力

標準出力に，必要な操作回数の最小値を表す整数を 1 行で出力せよ．

制限

すべての入力データは以下の条件を満たす.

- $3 \leqq N \leqq 300\,000$.
- $1 \leqq D_i \leqq 1\,000\,000\,000$.

小課題

小課題 1 [10 点]

- $N \leqq 8$ を満たす.

小課題 2 [20 点]

- $N \leqq 20$ を満たす.

小課題 3 [15 点]

- $N \leqq 5\,000$ を満たす.

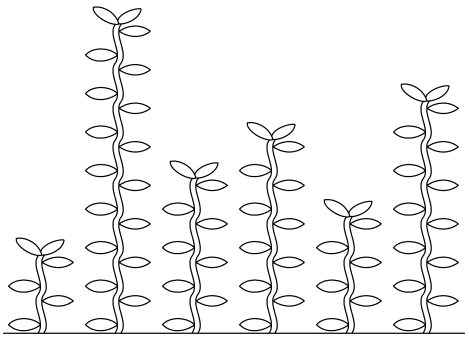
小課題 4 [55 点]

追加の制限はない.

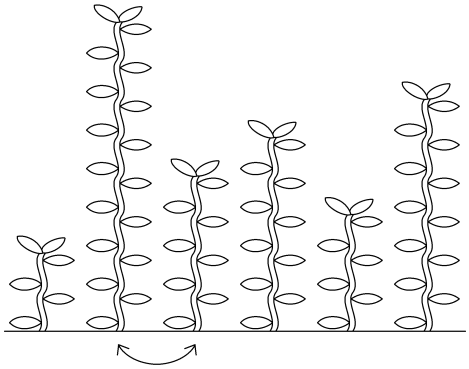
入出力例

入力例 1	出力例 1
6 2 8 4 5 3 6	3

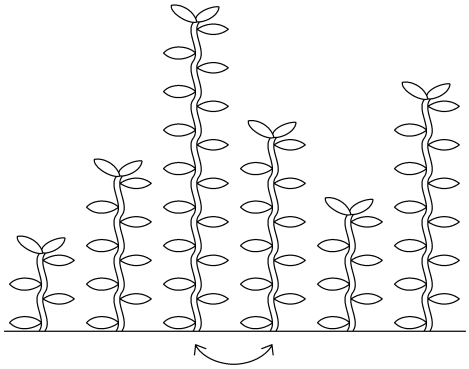
初期状態で IOI 草は下図のような配置となっている．



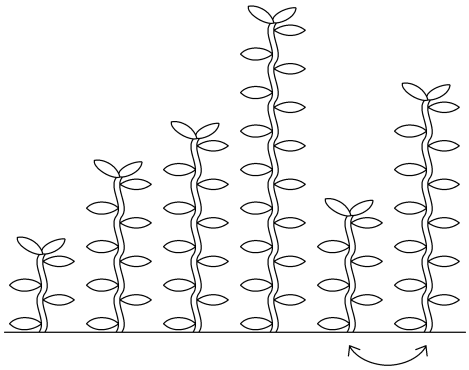
例えば，次のように動かすと，3 回の操作で IOI 草が枯れない配置にすることができる．



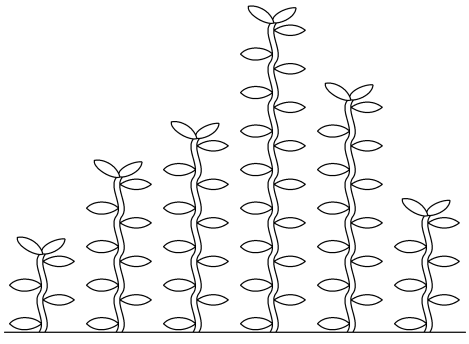
区画 2 と区画 3 の IOI 草を入れ替える．



区画 3 と区画 4 の IOI 草を入れ替える．



区画 5 と区画 6 の IOI 草を入れ替える．



この配置は，IOI 草が枯れない配置である．

入力例 2	出力例 2
5 4 4 2 4 4	2

区画 3 にある IOI 草を区画 1 または区画 5 に動かせば良い.

入力例 3	出力例 3
4 1 3 4 2	0

この入力例において，IOI 草を入れ替える操作をする必要はない.

歴史の研究 (Historical Research)

IOI 国の歴史の研究の第一人者であるジョイ教授のもとに，古代の IOI 国の住民が書いたとされる日記が届いた．ジョイ教授はこの日記から古代の IOI 国における生活を研究するため，日記に書かれている出来事を調査することにした．

この日記では N 日間に起きた出来事がそれぞれ 1 日あたり 1 つずつ書かれている．出来事は何種類かに分類されている．i 日目 $(1 \leqq i \leqq N)$ の出来事の種類は整数 X_i で表される．X_i の値が大きいほど規模が大きい出来事であると考えられている．

ジョイ教授は次のような方法で日記を分析することにした．

1. 日記の N 日間中で連続する何日間かを，分析する期間として選ぶ．
2. 出来事の種類 t の**重要度**を，$t \times$(その期間における種類 t の出来事の個数) とする．
3. すべての出来事の種類に対して重要度を計算し，その最大値を算出する．

あなたは，ジョイ教授から分析のためのプログラムを作ることを命じられた．このプログラムでは，分析する期間が与えられたときに重要度の最大値を求められるようにしたい．

課題

日記の N 日間の出来事の種類と，日記の中での期間を表すクエリが Q 個与えられたとき，それぞれのクエリについて出来事の重要度の最大値を求めるプログラムを作成せよ．

入力

標準入力から以下のデータを読み込め．

- 1 行目には，整数 N, Q が空白を区切りとして書かれている．これは，日記が N 日間分あり，クエリが Q 個与えられることを表す．
- 次の行には，N 個の整数 $X_1, \ldots, X_N$ が空白を区切りとして書かれており，X_i $(1 \leqq i \leqq N)$ は i 日目の出来事の種類を表す．
- 続く Q 行のうちの j 行目 $(1 \leqq j \leqq Q)$ には，整数 A_j, B_j $(1 \leqq A_j \leqq B_j \leqq N)$ が空白を区切りとして書かれており，j 番目のクエリが A_j 日目から B_j 日目までの期間に対するものであることを表す．

出力

標準出力に Q 行出力せよ．j 行目 $(1 \leqq j \leqq Q)$ に，j 番目のクエリに対する重要度の最大値を表す整数を出力せよ．

制限

すべての入力データは以下の条件を満たす.

- $1 \leqq N \leqq 100\,000$.
- $1 \leqq Q \leqq 100\,000$.
- $1 \leqq X_i \leqq 1\,000\,000\,000$ $(1 \leqq i \leqq N)$.

小課題

小課題 1 [5 点]

以下の条件を満たす.

- $N \leqq 100$.
- $Q \leqq 100$.

小課題 2 [10 点]

以下の条件を満たす.

- $N \leqq 5\,000$.
- $Q \leqq 5\,000$.

小課題 3 [25 点]

- $A_i \leqq A_j \leqq B_j \leqq B_i$ なる i, j $(1 \leqq i \leqq Q,\ 1 \leqq j \leqq Q,\ i \neq j)$ は存在しない.

小課題 4 [60 点]

追加の制限はない.

入出力例

入力例 1	出力例 1
5 5 9 8 7 8 9 1 2 3 4 4 4 1 4 2 4	9 8 8 16 16

- この日記は 5 日間からなり，日記に書かれている出来事の種類は 7, 8, 9 のいずれかである．
- 1 日目から 2 日目において，出来事の種類 7 の重要度は $7\times0=0$，出来事の種類 8 の重要度は $8\times1=8$，出来事の種類 9 の重要度は $9\times1=9$ であるから，これらのうちの最大値は 9 である．
- 3 日目から 4 日目において，出来事の種類 7 の重要度は $7\times1=7$，出来事の種類 8 の重要度は $8\times1=8$，出来事の種類 9 の重要度は $9\times0=0$ であるから，これらのうちの最大値は 8 である．
- 4 日目において，出来事の種類 7 の重要度は $7\times0=0$，出来事の種類 8 の重要度は $8\times1=8$，出来事の種類 9 の重要度は $9\times0=0$ であるから，これらのうちの最大値は 8 である．
- 1 日目から 4 日目において，出来事の種類 7 の重要度は $7\times1=7$，出来事の種類 8 の重要度は $8\times2=16$，出来事の種類 9 の重要度は $9\times1=9$ であるから，これらのうちの最大値は 16 である．
- 2 日目から 4 日目において，出来事の種類 7 の重要度は $7\times1=7$，出来事の種類 8 の重要度は $8\times2=16$，出来事の種類 9 の重要度は $9\times0=0$ であるから，これらのうちの最大値は 16 である．

入力例 2	出力例 2
8 4 9 9 19 9 9 15 9 19 1 4 4 6 3 5 5 8	27 18 19 19

この入力は小課題 3 の制約を満たしている．

入力例 3	出力例 3
12 15 15 9 3 15 9 3 3 8 16 9 3 17 2 7 2 5 2 2 1 12 4 12 3 6 11 12 1 7 2 6 3 5 3 10 7 10 1 4 4 8 4 8	18 18 9 30 18 15 17 30 18 15 18 16 30 15 15

ラーメンの食べ比べ (Ramen)

JOI 君と IOI ちゃんはラーメンが好きである．JOI 君はあっさりしたラーメンが好きなのに対して，IOI ちゃんはこってりしたラーメンが好きである．JOI 君と IOI ちゃんが住んでいる町には，ラーメン屋 0 からラーメン屋 $N-1$ までの N 軒のラーメン屋がある．

どのラーメン屋がこってりしたラーメンを提供しているか，またどのラーメン屋があっさりしたラーメンを提供しているかはわからない．そこで，JOI 君と IOI ちゃんは，近所のラーメン屋を巡って，最もあっさりしたラーメンを提供しているラーメン屋と最もこってりしたラーメンを提供しているラーメン屋がどこであるかを決めることにした．

JOI 君と IOI ちゃんが住んでいる町にあるラーメン屋には，そのラーメン屋で食べることができるラーメンの**こってり度**がそれぞれ定まっている．こってり度は 0 以上 $N-1$ 以下の整数であり，それぞれのラーメン屋のこってり度は互いに異なる．JOI 君と IOI ちゃんは 1 日に 2 軒のラーメン屋を巡り，味を比べることで，2 軒のラーメン屋のうちこってり度が高いのがどちらのラーメン屋であるかを判定することができる．

健康のため，JOI 君と IOI ちゃんはラーメンの食べ比べを行う日数を 600 日までに抑えたい．

課題

町にあるラーメン屋の軒数 N が与えられたとき，高々 600 日の食べ比べによって，N 軒のラーメン屋の中でこってり度が最も低いラーメン屋と，こってり度が最も高いラーメン屋を決定するプログラムを作成せよ．

実装の詳細

あなたは，こってり度が最も低いラーメン屋と，こってり度が最も高いラーメン屋を，高々 600 日の食べ比べによって決定する方法を実装した 1 個のプログラムを書かねばならない．
プログラムは，以下のルーチンを実装しなければならない．

- `void Ramen(int N)`

  このルーチンは，各テストケースに対し 1 回だけ呼び出される．引数 `N` は町にあるラーメン屋の軒数である．

  このルーチンは，`Compare` を呼び出すことによって，こってり度が最も低いラーメン屋とこってり度が最も高いラーメン屋を決定し，`Answer` を呼び出すことで終了しなければならない．

プログラム中では以下の関数を呼び出すことができる．

- `int Compare(int X, int Y)`

  この関数は，ラーメンの食べ比べを行う際に呼び出す．引数 `X`, `Y` はそれぞれ食べ比べを行うラーメン屋の番号 X, Y である．X と Y は 0 以上 $N-1$ 以下の互いに異なる整数でなくてはならない．これを満たさない引数とともに `Compare` を呼び出した場合は **不正解 [1]** となり，プログラムは終了する．

  この関数は，ラーメン屋 X のこってり度がラーメン屋 Y のこってり度より大きいときは 1 を返し，ラーメン屋 X のこってり度がラーメン屋 Y のこってり度より小さいときは -1 を返す．

  600 回より多く `Compare` を呼び出した場合は **不正解 [2]** となり，プログラムは終了する．

`Ramen` は，次のルーチンを呼び出すことによって終了しなければならない．`Ramen` が `Answer` を呼び出さなかった場合は **不正解 [3]** となり，プログラムは終了する．

- `void Answer(int X, int Y)`

  このルーチンは，ラーメンの食べ比べによって，こってり度が最も低いラーメン屋とこってり度が最も高いラーメン屋が定まっている状態で呼び出す．引数 `X`, `Y` は，それぞれ，こってり度が最も低いラーメン屋の番号 X と，こってり度が最も高いラーメン屋の番号 Y である．X, Y はそれぞれ 0 以上 $N-1$ 以下の整数でなくてはならない．これを満たさない引数とともに `Answer` を呼び出した場合は **不正解 [4]** となる．

  `Compare` の呼び出しの結果と矛盾しない答えが唯一であり，その答えを引数にして `Answer` を呼び出した場合は**正解**となる．そうでない場合は**不正解 [5]** となる．

  このルーチンが呼ばれると，プログラムは終了する．

採点上の注意

採点の際には，`Compare` の呼び出しの結果に矛盾しない答えが一意に定まっていない場合は，`Answer` の引数によらず，**不正解 [5]** と判定される．
採点用のデータの中には，`Compare` の返り値が以前の `Compare` の呼び出しの内容によって変化するものがある．この場合も `Compare` の返り値は以前の `Compare` の呼び出しの結果と矛盾することはない．

コンパイル・実行の方法

作成したプログラムをテストするための，採点プログラムのサンプルが，コンテストサイトからダウンロードできるアーカイブの中に含まれている．このアーカイブには，提出しなければならないファイルのサンプルも含まれている．

採点プログラムのサンプルは 1 つのファイルからなる．そのファイルは `grader.c` または `grader.cpp` である．例えば，作成したプログラムを `Ramen.c` または `Ramen.cpp` とするとき，作成したプログラムをテストするには， 次のようにコマンドを実行する．

- C の場合

  ```
  gcc -O2 -lm grader.c Ramen.c -o grader
  ```

- C++ の場合

  ```
  g++ -O2 grader.cpp Ramen.cpp -o grader
  ```

コンパイルが成功すれば， `grader` という実行ファイルが生成される．

実際の採点プログラムは，採点プログラムのサンプルとは異なることに注意すること．採点プログラムのサンプルは単一のプロセスとして起動する．このプログラムは，標準入力から入力を読み込み，標準出力に結果を出力する．

採点プログラムのサンプルの入力

採点プログラムのサンプルは標準入力から以下の入力を読み込む．

- 1 行目には整数 N, T が空白を区切りとして書かれている．N はラーメン屋の軒数である．採点プログラムのサンプルは，$T = 1$ であるデータを扱う．
- 続く N 行のうちの $i + 1$ 行目 $(0 \leqq i \leqq N - 1)$ には整数 A_i $(0 \leqq A_i \leqq N - 1)$ が書かれている．これは，ラーメン屋 i のこってり度を表す．

採点プログラムのサンプルの出力

プログラムの実行が正常に終了した場合，採点プログラムのサンプルは標準出力へ以下の情報を 1 行で出力する (引用符は実際には出力されない)．

- 正解の場合，“`Accepted`” と出力される．
- 不正解の場合，不正解の種類が “`Wrong Answer [2]`” のように出力される．

なお，採点プログラムのサンプルは，$A_X = 0$, $A_Y = N - 1$ なる X, Y について `Answer(X, Y)` を呼び出した場合は，**不正解 [5]** に該当する場合でも**正解**と判定する．これは実際の採点プログラムの挙動とは異なるので注意せよ．

制限

すべての入力データは以下の条件を満たす.

- $1 \leqq N \leqq 400$.

小課題

小課題 1 [20 点]

- $N \leqq 30$ を満たす.

小課題 2 [30 点]

- $N \leqq 300$ を満たす.

小課題 3 [50 点]

追加の制限はない.

やりとりの例

採点プログラムのサンプルが読み込む入力の例と，それに対応する関数とルーチンの呼び出しの例を以下に示す.

入力例	関数とルーチンの呼び出しの例	
3 1	呼び出し	戻り値
1	Compare(0, 1)	-1
2	Compare(0, 2)	1
0	Answer(2, 1)	プログラムは終了する