Zadanie: ARC
Architekci

XVI OI, etap II, dzień drugi. Plik źródłowy `arc.*` Dostępna pamięć: 32 MB. *12.02.2009*

Król Bajtazar postanowił wybudować sobie nowy pałac. Ogłosił więc konkurs na najlepszy projekt architektoniczny pałacu. Chcąc zmotywować architektów do spiesznej pracy, ogłosił też, że projekty będzie rozpatrywał w takiej kolejności, w jakiej będą nadsyłane.

Z tym zleceniem wiąże się ogromny prestiż, dlatego też architekci z całego świata nadsyłają swoje propozycje do królewskiej kancelarii. Projektów nadchodzi bardzo dużo, a Bajtazar nie ma czasu ich wszystkich przeglądać. Zrezygnował zatem z samodzielnego wykonywania tej czynności i poprosił swojego kanclerza o to, by wstępnie przejrzał nadchodzące projekty zgodnie z następującymi zasadami:

- kanclerz ma wybrać k projektów, odrzucając resztę od razu — Bajtazar wie, że więcej niż k projektów i tak nie będzie w stanie przejrzeć;
- projekty mają zostać przedstawione Bajtazarowi w takiej kolejności, w jakiej zostały nadesłane — w takiej też kolejności Bajtazar będzie je przeglądał, zgodnie z tym co ogłosił;
- spośród wszystkich ciągów k projektów spełniających powyższe warunki, kanclerz ma wybrać ciąg **najlepszy**, zgodnie z poniższą definicją.
  Powiemy, że ciąg projektów $(p_1, p_2, \ldots, p_k)$ jest **lepszy** od ciągu $(r_1, r_2, \ldots, r_k)$, jeśli dla pewnego $l \geqslant 1$ pierwszych $l-1$ projektów w obu ciągach jest równie dobrych, zaś l-ty projekt w ciągu p jest lepszy od l-tego projektu w ciągu r (czyli $p_i = r_i$ dla $i < l$ i $p_l > r_l$).

Projekty cały czas nadchodzą i nie wiadomo, do kiedy Bajtazar rozkaże je przyjmować. Kanclerz nie chce zostawiać wyboru k projektów na ostatni moment, jednak bardzo boi się popełnić błąd i narazić na gniew króla. Dlatego poprosił Cię o pomoc.

Napisz program, który:

- pobierze za pomocą dostarczonej biblioteki liczbę k oraz ciąg liczb reprezentujących jakość kolejnych projektów,
- wyznaczy najlepszy ciąg k projektów, zgodnie z podanymi zasadami,
- zwróci jakości wybranych projektów za pomocą biblioteki.

Opis użycia biblioteki

Aby użyć biblioteki, należy wpisać w swoim programie:

- C/C++: `#include ''carclib.h''`
- Pascal: `uses parclib;`
- Java: nic nie trzeba robić, lecz by uruchomić rozwiązanie, należy mieć skompilowaną bibliotekę jarclib (plik `jarclib.class`) w tym samym katalogu co program.

Biblioteka udostępnia trzy procedury, funkcje lub metody statyczne:

- `inicjuj` — zwraca liczbę całkowitą k ($1 \leqslant k \leqslant 1\,000\,000$), określającą, jak wiele projektów ma zawierać ciąg wynikowy. Powinna być użyta dokładnie raz, na samym początku działania programu.
  - C/C++: `int inicjuj();`
  - Pascal: `function inicjuj(): longint;`
  - Java: `public static int inicjuj();`, będąca metodą statyczną klasy `jarclib`.
- `wczytaj` — i-te wywołanie zwraca liczbę całkowitą p_i ($1 \leqslant p_i \leqslant 1\,000\,000\,000$) oznaczającą jakość i-tego projektu (im większa liczba, tym lepszy projekt), albo 0, co oznacza, że nie ma już więcej projektów. Liczba projektów nie jest znana przed wczytaniem danych, jednak możesz założyć, że wszystkich projektów jest przynajmniej k, a co najwyżej $15\,000\,000$. Funkcja ta powinna być wywoływana do momentu, aż skończą się projekty, i **ani razu więcej**.
  - C/C++: `int wczytaj();`
  - Pascal: `function wczytaj(): longint;`
  - Java: `public static int wczytaj();`, będąca metodą statyczną klasy `jarclib`.
- `wypisz` — za pomocą tej procedury/funkcji wypisujesz jakości kolejnych projektów, które kanclerz przedstawi królowi. Powinna być ona użyta dokładnie k razy; w i-tym wywołaniu należy podać jakość i-tego w kolejności projektu. k-te wywołanie tej procedury/funkcji zakończy działanie Twojego programu.

- C/C++: `void wypisz(int jakoscProjektu);`
- Pascal: `procedure wypisz(jakoscProjektu: longint);`
- Java: `public static void wypisz(int jakoscProjektu);`, będąca metodą statyczną klasy `jarclib`.

Twój program nie może otwierać żadnych plików ani używać standardowego wejścia i wyjścia. Rozwiązanie będzie kompilowane wraz z biblioteką za pomocą następujących poleceń:

- C: `gcc -O2 -static carclib.c arc.c -lm`
- C++: `g++ -O2 -static carclib.c arc.cpp -lm`
- Java: `javac arc.java`, a skompilowany plik biblioteki `jarclib` — `jarclib.class` — będzie się znajdował w tym samym katalogu.
- Pascal: `ppc386 -O2 -XS -Xt arc.pas`, a plik biblioteki `parclib` będzie znajdował się w tym samym katalogu.

W katalogu ??? możesz znaleźć przykładowe pliki bibliotek i przykładowe rozwiązania ilustrujące sposób ich użycia. Przykładowa biblioteka wczytuje scenariusz testowy ze standardowego wejścia w następującym formacie:

- Pierwszy wiersz wejścia zawiera jedną dodatnią liczbę całkowitą k.
- Kolejne wiersze wejścia zawierają po jednej dodatniej liczbie całkowitej; $(i+1)$-szy wiersz zawiera liczbę p_i, oznaczającą jakość i-tego zgłoszonego projektu.
- Ostatni wiersz wejścia zawiera liczbę 0, oznaczającą koniec listy projektów.

Przykładowa biblioteka wypisuje na standardowe wyjście k wierszy — jakości projektów zgłoszone przez program.

Przykładowy przebieg programu

C/C++	Pascal	Java	Zwracane wartości i wyjaśnienia
`k = inicjuj();`	`k := inicjuj();`	`k = jarclib.inicjuj();`	Od tego momentu $k = 3$.
			Wczytywanie jakości kolejnych projektów.
`wczytaj();`	`wczytaj();`	`jarclib.wczytaj();`	12
`wczytaj();`	`wczytaj();`	`jarclib.wczytaj();`	5
`wczytaj();`	`wczytaj();`	`jarclib.wczytaj();`	8
`wczytaj();`	`wczytaj();`	`jarclib.wczytaj();`	3
`wczytaj();`	`wczytaj();`	`jarclib.wczytaj();`	15
`wczytaj();`	`wczytaj();`	`jarclib.wczytaj();`	8
`wczytaj();`	`wczytaj();`	`jarclib.wczytaj();`	0 — oznacza koniec ciągu projektów
			Wypisujemy rozwiązanie, czyli 3-elementowy ciąg:
`wypisz(12);`	`wypisz(12);`	`jarclib.wypisz(12);`	12
`wypisz(15);`	`wypisz(15);`	`jarclib.wypisz(15);`	15
`wypisz(8);`	`wypisz(8);`	`jarclib.wypisz(8);`	8