MALACHOVIACUS
INFORMATICUS

Płock, 21 marca 2015 r.

Instrukcje dla zawodników

Arkusze otwieramy na wyraźne polecenie komisji. Wszystkie poniższe instrukcje zostaną odczytane i wyjaśnione.

1. Arkusz składa się z **3 zadań**.
2. Każde zadanie składa się z **wprowadzenia** oraz kilku **pytań**.
3. Liczba punktów możliwa do uzyskania za każde pytanie podana jest przy jego treści. Suma tych punktów w każdym zadaniu wynosi 20.
4. Przed udzieleniem odpowiedzi na pytania **przeczytaj dokładnie** wprowadzenie oraz treści poleceń.
5. Swoje odpowiedzi zapisz **czytelnie** na przeznaczonych do tego arkuszach.
6. Do zapisu odpowiedzi używaj wyłącznie **długopisu lub pióra** z **czarnym lub niebieskim** tuszem. Do wykonywania rysunków możesz użyć ołówka.
7. Każdy arkusz odpowiedzi powinien zawierać odpowiedź, lub jej część, na **tylko jedno** zadanie.
8. Na pierwszej stronie każdego arkusza odpowiedzi, w prawym górnym rogu, zapisz czytelnie **swój kod** oraz **numer zadania**.
9. Czas na rozwiązanie zadań wynosi **180 minut**.

Powodzenia!

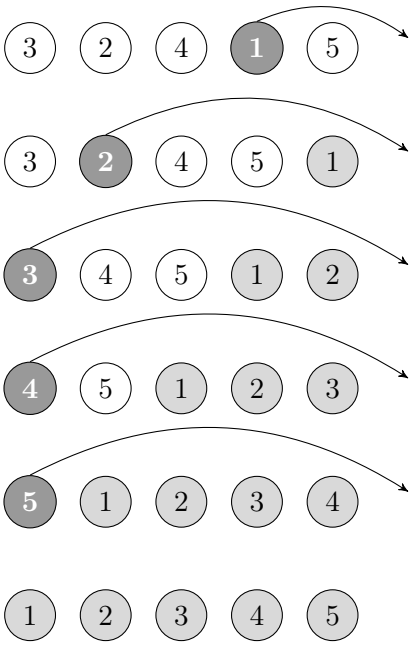
Zadanie 1

Sortowanie przez wybieranie

Jednym z podstawowych problemów w informatyce jest problem **sortowania**. Zadanie polega na uporządkowaniu pewnego zestawu liczb w kolejności rosnącej, to jest takiej, że każda następna liczba jest większa od poprzedniej. Na przykład zestaw liczb 3, 2, 4, 1, 5 po posortowaniu wyglądać będzie tak: 1, 2, 3, 4, 5.

Wydaje się, że jest to proste zadanie, jednakże komputer nie jest w stanie po prostu spojrzeć na liczby i wybrać odpowiedniej kolejności. Ty też możesz mieć problemy, jeżeli liczb nie jest kilka lub kilkanaście, ale kilka tysięcy lub milionów.

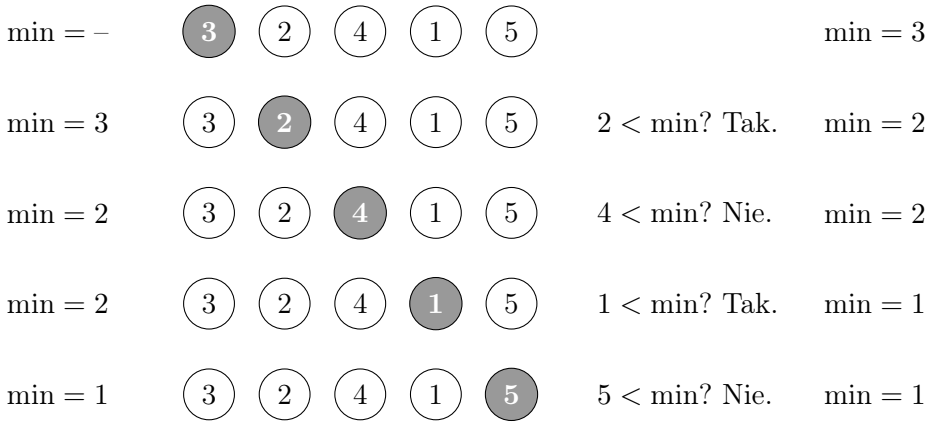
Jednym z najprostszych rozwiązań tego problemu jest **sortowanie przez wybieranie**. Polega ono na ciągłym wybieraniu **minimum** (najmniejszej liczby) i usuwaniu go z zestawu, aż nie zostanie nam żadna liczba. Kolejno wybrane wartości będą tworzyć nasz posortowany zestaw. Przykład na rysunku 1.



Rysunek 1: Przykład sortowania przez wybieranie. Kolorem ciemnoszarym oznaczone są kolejne minima, jasnoszarym zaś posortowane już liczby.

Pozostaje nam problem wybrania najmniejszej spośród liczb. Posiada on jednak równie proste rozwiązanie. Wystarczy przejrzeć wszystkie elementy zestawu jeden po drugim, zapamiętując jaka była najmniejsza liczba jaką do tej pory widzieliśmy. Jeżeli obecna wartość jest mniejsza od tej zapamiętanej, zapamiętujemy ją jako nowe minimum. Przykład na rysunku 2.

Istotną cechą każdego algorytmu jest liczba operacji, które składają się na jego wykonanie w zależności od liczby i wielkości danych liczb. W tym zadaniu szczególnie interesuje nas liczba potrzebnych porównań. W naszym przykładzie wyszukiwania minimum musimy wykonać ich 4 (patrz rysunek 2).



Rysunek 2: Przykład wybierania minimum. Kolorem ciemnoszarym oznaczona jest obecnie rozpatrywana wartość. Z lewej strony znajduje się wartość zapamiętanego minimum przed rozpatrzeniem kolejnej wartości, a z prawej strony – po rozpatrzeniu.

Pytania

Pytanie 1 Dany jest zestaw liczb: 9, 7, 10, 5, 8, 6, 2, 4. Jakie będą kolejne zapamiętane wartości dotychczasowego minimum, jeżeli będziemy próbowali znaleźć najmniejszą liczbę tego zestawu używając algorytmu z treści zadania? [3 punkty]

Pytanie 2 Ile porównań należy wykonać, aby znaleźć najmniejszą spośród 8 liczb? 100 liczb? n liczb? [4 punkty]

Pytanie 3 Zamiast przeszukiwać zestaw liczb po kolei, możemy znaleźć minimum tzw. systemem pucharowym. Dobieramy wszystkie liczby w pary, porównujemy je i pozostawiamy tylko te mniejsze z każdej pary. Powtarzamy tak długo aż zostanie nam tylko jedna liczba – nasze minimum.

Ile porównań będzie wymagać znalezienie minimum n liczb w ten sposób? Uzasadnij. Możesz założyć, że n jest potęgą dwójki (będzie się dzielić na 2 tak długo aż otrzymamy 1). [4 punkty]

Pytanie 4 Chcemy posortować zestaw n liczb używając sortowania przez wybieranie. Ile razy będziemy musieli wykonać operację wyszukania minimum ciągu? Uzasadnij. [3 punkty]

Używając sposobu wyszukiwania minimum opisanego we wprowadzeniu, ile porównań będziemy musieli wykonać sortując zestaw n liczb? Uzasadnij. [3 punkty]

Ile razy, szacunkowo, wzrośnie wymagana liczba porównań, jeżeli zwiększymy liczbę liczb w zestawie dwukrotnie? Uzasadnij. [3 punkty]

Do udzielenia odpowiedzi na pytania mogę Ci się przydać poniższe wzory:
$1 + 2 + 3 + 4 + \cdots + n = \frac{n(n+1)}{2}$ $\quad$ $1 + 2 + 4 + 8 + \cdots + 2^n = 2^{n+1} - 1$

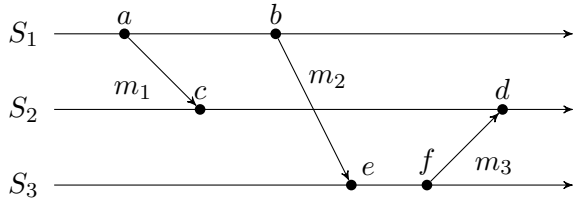
ZADANIE 2

Zgodne przekroje

Duże serwisy internetowe, jak Google czy Facebook, obsługiwane są przez wiele serwerów jednocześnie. Serwery te muszą się ze sobą komunikować, aby każdy z nich był w stanie przekazywać użytkownikom aktualne dane. Istotne jest także to, aby serwery posiadały te same informacje co do kolejności zmian w bazie danych. Nie jest to jednak wcale takie proste. W tym zadaniu przyjrzymy się kwestii mocno związanej z tym problemem – zgodnym przekrojom.

W naszym uproszczonym scenariuszu mamy kilka **serwerów**, oznaczonych przez S_1, S_2 itd. Każdy z nich posiada swoją **lokalną oś czasu**, na której umieszczone są **wydarzenia**. Na diagramach wydarzenia oznaczamy czarnymi kropkami na osiach czasu, z dodanymi oznaczeniami literowymi dla ich rozróżnienia.

Serwery mogą wysyłać sobie **wiadomości**. Do każdej z nich przypisane są dwa wydarzenia: wysyłka i odbiór. Oznaczamy je tak jak wszystkie inne wydarzenia, ale dodatkowo łączymy je strzałkami, dla określenia kierunku przesyłu wiadomości. Każda z wiadomości będzie również oznaczona przez m_i, gdzie i to dowolna liczba naturalna.



Mówimy, że wydarzenie x **wydarzyło się przed** y, jeżeli zachodzi jeden z poniższych warunków:

1. x i y wydarzyły się na tym samym serwerze i x znajduje się wcześniej niż y na lokalnej osi czasu. Na diagramie powyżej, na podstawie tej reguły, zachodzi $a \to b$, $c \to d$ i $e \to f$.

2. x jest wysyłką, a y odbiorem tej samej wiadomości. Stąd na diagramie powyżej mamy, na przykład, $b \to e$ czy $f \to d$.

3. istnieje jakieś wydarzenie z takie że $x \to z$ i $z \to y$. Stąd mamy $b \to f$, ponieważ istnieje e takie, ze $b \to e$ (reguła 2) oraz $e \to f$ (reguła 1). Zwróć uwagę, że na podstawie tej reguły zachodzi również $b \to d$, ponieważ istnieje f, takie że $b \to f$ (patrz wyżej) oraz $f \to d$ (reguła 2).

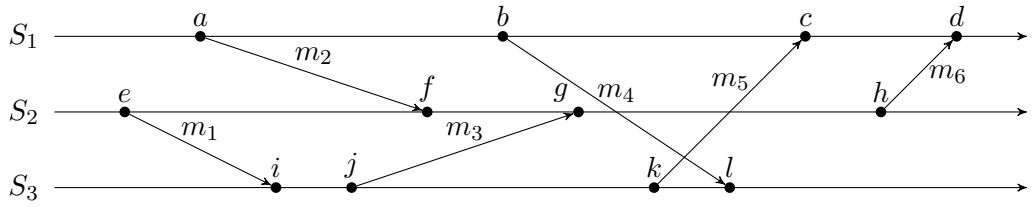
Możliwe jest, że dla dwóch wydarzeń x i y nie zachodzi ani $x \to y$ ani $y \to x$. Mówimy wtedy, że x i y **zachodzą równolegle**, co zapisujemy $x \sim y$. Na diagramie powyżej mamy, na przykład, $b \sim c$ czy $c \sim f$.

Przekrojem zestawu wydarzeń będziemy nazywać dowolny ich podział na dwie części. Każde wydarzenie musi znajdować się po jednej, i tylko jednej, stronie przekroju. Przekrojem wydarzeń z diagramu powyżej będzie na przykład $a, b, c/d, e, f$, czy $b, d/a, c, e, f$, ale także $a, b, c, d, e, f/-$. Nie jest istotna kolejność

wydarzeń w każdej z części, stąd przekrój $a, b, c/d, e, f$ jest identyczny z przekrojem $b, a, c/f, e, d$. Istotna jest za to kolejność części, stąd $a, b, c/d, e, f$ i $d, e, f/a, b, c$ to dwa różne przekroje.

Zgodnym przekrojem będziemy nazywać taki przekrój, w którym nie ma takiej pary wydarzeń x i y, że $x \to y$ oraz x znajduje się po prawej, a y po lewej stronie przekroju. Przykładem zgodnego przekroju z diagramu powyżej będzie $a, b, c/d, e, f$, ale już nie $a, b, c, d/e, f$, gdyż $f \to d$.

Wszystkie pytania w tym zadaniu odnoszą się do poniższego diagramu:



PYTANIA

PYTANIE 1 Które z poniższych zależności są prawdziwe, a które fałszywe?

(a) $b \to l$ [1 punkt]

(b) $a \leftarrow b$ [1 punkt]

(c) $j \to f$ [1 punkt]

PYTANIE 2 Uzupełnij $\leftarrow$ lub $\to$. Podaj reguły, na podstawie których zachodzi każda z zależności. W przypadku użycia reguły 3 podaj oraz uzasadnij także zależności, z których korzystasz.

(a) $e \ldots g$ [2 punkty]

(b) $c \ldots j$ [2 punkty]

(c) $d \ldots i$ [3 punkty]

PYTANIE 3 Podaj przykład zgodnego przekroju zawierającego wydarzenie k po lewej stronie. [2 punkty]

Znajdź taki zgodny przekrój, który zawiera wydarzenie k po **lewej** stronie, a do tego zawiera jak najmniej wydarzeń po lewej stronie. Dlaczego nie może istnieć taki przekrój posiadający mniej wydarzeń po lewej stronie?
[4 punkty]

Znajdź taki zgodny przekrój, który zawiera wydarzenie k po **prawej** stronie, a do tego zawiera jak najmniej wydarzeń po prawej stronie. Dlaczego nie może istnieć taki przekrój posiadający mniej wydarzeń po prawej stronie?
[4 punkty]

Zadanie 3

Rachunek lambda

Rachunek lambda jest systemem matematycznym pozwalającym na opisywanie funkcji oraz rozumowanie o nich. Został on sformułowany w 1930 roku przez amerykańskiego matematyka Alonzo Churcha. Jego nazwa pochodzi od greckiej litery lambda (λ) dość często pojawiającej się w zapisie rachunku.

W rachunku lambda posługujemy się **wyrażeniami lambda**. Wyrażenia oznaczamy wielkimi literami alfabetu, na przykład M, N czy K. Wyróżniamy trzy rodzaje wyrażeń: **wartości**, **aplikacje** i **abstrakcje**.

Wartość to dowolna liczba lub pojedyncza mała litera alfabetu. Przykładami wartości są 15, 42, x czy a.

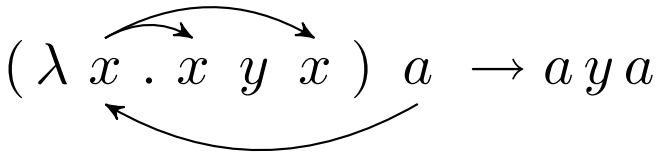
Aplikacja to dowolna para wyrażeń $M\,N$, na przykład $a\,x$. Nie muszą być to jednak dwie wartości. Każde z dwóch wyrażeń M, N może być dowolnym wyrażeniem lambda, tj. wartością, aplikacją lub abstrakcją. Zapis $M\,N$ oznacza „przekaż wyrażenie N wyrażeniu M".

Abstrakcje mają postać $\lambda x.M_x$. Wartość x nazywamy **parametrem** abstrakcji. Wyrażenie M_x jest dowolnym wyrażeniem lambda wykorzystującym x. Nazywamy je **wyrażeniem pod abstrakcją**. Zapis $\lambda x.M_x$ oznacza „przyjmij wyrażenie x i zwróć jako wynik wyrażenie M_x". Przykładem abstrakcji jest wyrażenie $\lambda x.a\,x$, które mając daną wartość x zwraca wyrażenie $a\,x$.

Oczywiście możemy zastąpić x dowolną inną literą, o ile nie koliduje ona z pozostałymi oznaczeniami wewnątrz wyrażenia. Na przykład wyrażenie $\lambda y.a\,y$ jest równoważne z wyrażeniem $\lambda x.a\,x$, natomiast wyrażenie $\lambda a.a\,a$ nie jest równoważne z wyrażeniem $\lambda x.a\,x$.

Do zapisu wyrażeń potrzebujemy również, znanych Ci już, nawiasów. Na przykład wyrażenie $(\lambda x.x)\,a$, będące aplikacją, różni się od wyrażenia $\lambda x.(x\,a)$, będącego abstrakcją. Przyjmujemy, że zapis $\lambda x.x\,a$, rozumiemy jako $\lambda x.(x\,a)$. Podobnie przyjmujemy, że zapis $M\,N\,K$ oznacza $(M\,N)\,K$.

Wyrażenia lambda możemy **redukować**, pozbywając się z nich aplikacji. Redukcji podlegają aplikacje, których pierwszym wyrażeniem jest abstrakcja (np. $(\lambda x.a\,x)\,b$). W takim wyrażeniu możemy zastąpić całą aplikację wyrażeniem spod abstrakcji, jednocześnie zastępując w nim parametr drugim z wyrażeń aplikacji. Na przykład wyrażenie $(\lambda x.a\,x)\,b$ możemy zredukować do $a\,b$. Redukcję wyrażenia M do wyrażenia N zapisujemy symbolicznie $M \to N$.



Rysunek 3: Graficzne przedstawienie redukcji wyrażenia $(\lambda x.x\,y\,x)\,a$.

Drugie wyrażenie aplikacji nie musi być pojedynczą wartością – może to być dowolne wyrażenie lambda. Możemy na przykład napisać $(\lambda x.x\,y)(\lambda z.z)$, co zredukuje

się do $(\lambda z.z)\,y$. Wyrażenie pod abstrakcją też może być dowolne, w tym może ono być kolejną abstrakcją, na przykład $\lambda x.\lambda y.x\,y$, co rozumiemy jako $\lambda x.(\lambda y.(x\,y))$.

Może okazać się, że po dokonaniu redukcji otrzymamy wyrażenie, które ponownie możemy zredukować. Jeżeli dokonując kilku kolejnych redukcji wyrażenia M otrzymamy wyrażenie N, możemy zapisać symbolicznie $M \to^n N$.

Jeżeli otrzymane po redukcji wyrażenie nie może być dalej zredukowane, mówimy że znajduje się ono w **postaci normalnej**. Jeżeli $M \to^n N$, gdzie N jest w postaci normalnej, mówimy że N **jest postacią normalną** M. Znalezienie postaci normalnej danego wyrażenia sprowadza się do redukowania tego wyrażenia do momentu, kiedy nie będzie to już dłużej możliwe. Otrzymane w ten sposób wyrażenie będzie postacią normalną pierwotnego wyrażenia lambda.

Istnieją takie wyrażenia lambda, które nie posiadają postaci normalnej. Na przykład $(\lambda x.x\,x)(\lambda x.x\,x)$ po zredukowaniu ponownie staje się $(\lambda x.x\,x)(\lambda x.x\,x)$. Nie znajduje się ono w postaci normalnej (istnieje możliwa redukcja) i nie da się go do takiej zredukować (jedyna możliwa redukcja daje nam to samo wyrażenie).

Rachunek lambda możemy rozszerzyć o działania arytmetyczne, jak dodawanie czy mnożenie. Dzięki temu możemy zapisywać wyrażenia takie jak $\lambda x.x + 5$. Należy jednak pamiętać, że redukcja dotyczy tylko aplikacji. Stąd postacią normalną wyrażenia $(\lambda x.x + 5)\,2$ jest $2 + 5$, a nie 7.

Pytania

Pytanie 1 Co otrzymamy po **pojedynczym** zredukowaniu każdego z poniższych wyrażeń?

(a) $(\lambda x.2 \cdot x)\,5$ [1 punkt]

(b) $(\lambda x.\lambda y.x + y)\,3\,5$ [1 punkt]

(c) $(\lambda x.x)((\lambda z.z)u)$ [2 punkty]

Pytanie 2 Znajdź postać normalną każdego z poniższych wyrażeń.

(a) $(\lambda x.x)(\lambda x.x)$ [1 punkt]

(b) $(\lambda x.x\,y)(\lambda z.z\,u)$ [2 punkty]

(c) $(\lambda x.(\lambda y.x \cdot y)\,5)\,4$ [2 punkty]

(d) $(\lambda x.x\,(\lambda a.\lambda b.b)\,(\lambda c.\lambda d.c))\,(\lambda e.\lambda f.e)$ [3 punkty]

Pytanie 3 Spójrz na poniższe wyrażenie:

$$\lambda f.(\lambda x.f(x\,x))(\lambda x.f(x\,x))$$

Jak będzie wyglądać to wyrażenie po wykonaniu jednej redukcji? Dwóch redukcji? Czy to wyrażenie posiada postać normalną? Jeżeli tak, podaj ją, jeżeli nie, wyjaśnij dlaczego nie. [8 punktów]