MINISTERSTWO EDUKACJI NARODOWEJ
FUNDACJA ROZWOJU INFORMATYKI
KOMITET GŁÓWNY OLIMPIADY INFORMATYCZNEJ

XX OLIMPIADA INFORMATYCZNA

Drodzy Czytelnicy Delty,

przedstawiamy Wam zadania zawodów I stopnia jubileuszowej, XX Olimpiady Informatycznej. Tych z Was, którzy chcą wziąć udział w Olimpiadzie, zobowiązujemy do przeczytania „Zasad organizacji zawodów XX OI", w których zawarte są najważniejsze informacje o tym, w jaki sposób należy przygotować swoje rozwiązania. Rozwiązania należy zgłaszać przez stronę sio.mimuw.edu.pl, na której znajdują się także wybrane odpowiedzi na pytania zawodników, narzędzia do sprawdzania rozwiązań pod względem formalnym oraz forum służące do wymiany doświadczeń między zawodnikami.

Termin nadsyłania rozwiązań przedstawionych tu zadań I etapu upływa **12 listopada 2012 roku**.

Jubileuszowa edycja Olimpiady skłania do pewnych podsumowań. W dotychczasowych edycjach wystartowało łącznie 16 830 zawodników. W finałach Olimpiady wystąpiło 1105 zawodników, a tytuł laureata uzyskało 374 zawodników. Reprezentanci Polski zdobyli na Międzynarodowej Olimpiadzie Informatycznej łącznie 85 medali: 32 złote, 31 srebrnych i 22 brązowe, i dwukrotnie, w 2006 i 2007 roku, zwyciężali w tych zawodach (odpowiednio: Filip Wolski i Tomasz Kulczyński). W nieoficjalnej klasyfikacji drużynowej wszech czasów Polska znajduje się na czwartym miejscu tych zawodów – za Chinami, Rosją i Stanami Zjednoczonymi – a w pierwszej dziesiątce najlepszych zawodników w historii tych zawodów jest aż trzech Polaków (Filip Wolski – 2. miejsce, Andrzej Gąsienica-Samek – 4. miejsce, Marcin Andrychowicz – 6. miejsce).

W ciągu 19 lat istnienia Olimpiada wydała wiele materiałów edukacyjnych, do których zaliczają się książeczki zawierające opracowania zadań z kolejnych edycji Olimpiady oraz pięć książeczek ze zorganizowanych przez Polskę zawodów międzynarodowych, w tym tych najważniejszych – Międzynarodowej Olimpiady Informatycznej w 2005 roku. Materiały te są dostępne elektronicznie na oficjalnej stronie Olimpiady: oi.edu.pl. Istnieje także portal treningowy Olimpiady, main.edu.pl, na którym dostępne są kursy programowania i algorytmiki oraz możliwość rozwiązywania ponad 900 zadań z różnych zawodów informatycznych, w tym prawie wszystkich archiwalnych zadań z Olimpiady. Dziennie portal MAIN otrzymuje średnio kilkaset zgłoszeń, w tym od użytkowników zagranicznych – wiele zadań jest bowiem dostępnych także w angielskiej wersji językowej. Już od dziesięciu lat rozwiązania zawodników są sprawdzane za pomocą Systemu Internetowego Olimpiady (SIO), a w tej edycji Olimpiady zacznie być używany jego następca, system SIO2. Od niedawna działa portal treningowy dla uczniów i nauczycieli, szkopul.edu.pl.

Chciałbym jeszcze wspomnieć o tegorocznych sukcesach Polaków w zawodach międzynarodowych.

Wyjątkowo dobrym wynikiem Polaków zakończyła się Bałtycka Olimpiada Informatyczna (maj 2012) – Polacy zajęli całe podium! Trzy pierwsze miejsca zdobyli, i złote medale przywieźli do Polski, kolejno, zawodnicy: Krzysztof Pszeniczny, Szymon Stankiewicz i Karol Farbiś. Złoty medal uzyskał także Stanisław Dobrowolski, natomiast srebrne medale zdobyli Szymon Łukasz i Przemysław Jakub Kozłowski.

Na zakończonej w lipcu Olimpiadzie Informatycznej Krajów Europy Środkowej również wszyscy reprezentanci Polski zdobyli medale. Złoto (2. miejsce w łącznej klasyfikacji) uzyskał Mateusz Gołębiewski, srebrne medale przywieźli Wiktor Kuropatwa i Bartłomiej Dudek, a brąz zdobył Karol Farbiś.

W tym roku Międzynarodowa Olimpiada Informatyczna odbyła się wyjątkowo późno, bo pod koniec września. Nie przeszkodziło to reprezentantom Polski w osiągnięciu bardzo dobrych, a zarazem wyjątkowo równych, wyników. Z tych najbardziej prestiżowych zawodów dla licealistów złoty medal przywiózł Karol Farbiś (7. miejsce). Wszyscy pozostali członkowie reprezentacji – Wojciech Nadara, Wiktor Kuropatwa oraz Bartłomiej Dudek – zdobyli srebrne medale, w dwóch przypadkach ocierając się o złoto.

Ten rok okazał się dla Polaków bardzo udany także jeśli chodzi o zawody studenckie i zawody w kategorii „open". Poza wspomnianym już w *Delcie* wicemistrzostwem świata w programowaniu zespołowym, jakie zdobyła drużyna Uniwersytetu Warszawskiego w składzie: Tomasz Kulczyński, Jakub Pachocki, Wojciech Śmietanka, jeden z jej członków odniósł też znaczący triumf w zawodach indywidualnych. Jakub Pachocki zwyciężył w tegorocznym finale konkursu Google Code Jam w Nowym Jorku, pokonując w kolejnych rundach eliminacyjnych łącznie 20 tysięcy programistów z całego świata. Był to drugi tak dobry występ Polaka w historii tych zawodów; siedem lat wcześniej triumfował w nich Marek Cygan. W chwili oddawania tego tekstu do druku w Orlando na Florydzie trwają finały konkursu TopCoder Open 2012, w których Polskę reprezentują obaj wspomniani tu zwycięzcy zawodów Google Code Jam oraz Marcin Andrychowicz – trzymamy kciuki za ich występ!

Na koniec chciałbym podziękować wypróbowanym przyjaciołom Olimpiady: Fundacji Rozwoju Informatyki – formalnemu organizatorowi, współorganizatorowi – firmie Asseco Poland SA, firmie Combidata Poland, a także naszym partnerom: Ogólnopolskiej Fundacji Edukacji Komputerowej oraz Ośrodkowi Edukacji Informatycznej i Zastosowań Komputerów w Warszawie.

prof. dr hab. Krzysztof Diks
Przewodniczący Komitetu Głównego
Olimpiady Informatycznej

Zadania

CENNIK

plik źródłowy cen.*, dostępna pamięć 64 MB

Najpopularniejszym środkiem transportu w Bajtocji od zawsze była kolej. Spośród n miast tego kraju, m par miast jest połączonych odcinkami torów należącymi do Bajtockich Kolei Państwowych (BKP). Odcinki nie krzyżują się nigdzie poza miastami i mogą prowadzić tunelami lub mostami. Koszt przejazdu między dwoma miastami połączonymi bezpośrednio odcinkiem torów jest zawsze taki sam i wynosi a bajtalarów.

Obecnie sytuacja na rynku usług komunikacyjnych w Bajtocji uległa zmianie. Pojawiła się konkurencja dla BKP – powstały Bajtockie Linie Lotnicze (BLL). BLL zamierzają uruchomić połączenia lotnicze między niektórymi parami miast. Ponieważ jazda bajtocką koleją jest bardzo wygodna, zarząd BLL postanowił uruchamiać połączenia lotnicze tylko między takimi parami miast, dla których **nie** istnieje bezpośrednie połączenie kolejowe. Ze względów ekonomicznych, BLL utworzy połączenia lotnicze jedynie między tymi parami miast, dla których najtańsze połączenie kolejowe wymaga dokładnie jednej przesiadki. Koszt biletu lotniczego na jedno połączenie będzie stały i równy b bajtalarów.

Żeby pomóc mieszkańcom Bajtocji w planowaniu podróży, Ministerstwo Transportu Bajtocji (MTB) postanowiło stworzyć cenniki zawierające koszty najtańszych tras między poszczególnymi miastami kraju. **Trasę** rozumiemy tu jako sekwencję złożoną z dowolnej liczby pojedynczych połączeń kolejowych lub lotniczych. Zadanie stworzenia cenników przypadło w udziale Bajtazarowi, który pracuje jako urzędnik w MTB. Czy pomógłbyś mu w napisaniu programu, który wyznaczy odpowiednie cenniki?

Dodajmy dla jasności, że wszystkie połączenia kolejowe i lotnicze w Bajtocji są dwukierunkowe.

Wejście

Pierwszy wiersz standardowego wejścia zawiera pięć liczb całkowitych n, m, k, a oraz b ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 100\,000$, $1 \leqslant k \leqslant n$, $1 \leqslant a, b \leqslant 1\,000$) pooddzielanych pojedynczymi odstępami. Liczby n oraz m oznaczają odpowiednio liczbę miast oraz liczbę połączeń kolejowych w Bajtocji. Dla uproszczenia miasta w Bajtocji numerujemy od 1 do n. Kolejne liczby w wierszu oznaczają: k – numer miasta początkowego, dla którego należy wygenerować cennik opisujący najtańsze trasy; a – koszt biletu na jedno połączenie kolejowe; b – koszt biletu na jedno połączenie lotnicze.

Każdy z kolejnych m wierszy zawiera dwie liczby całkowite u_i oraz v_i ($1 \leqslant u_i, v_i \leqslant n$, $u_i \neq v_i$ dla $i = 1, 2, \ldots, m$) oddzielone pojedynczym odstępem, oznaczające numery miast połączonych bezpośrednim odcinkiem torów.

Możesz założyć, że z miasta numer k można dojechać koleją do wszystkich pozostałych miast kraju.

W testach wartych łącznie 30% punktów zachodzą dodatkowe warunki $n \leqslant 700$ oraz $m \leqslant 700$.

Wyjście

Twój program powinien wypisać na standardowe wyjście n wierszy. Wiersz o numerze i (dla $i = 1, 2, \ldots, n$) powinien zawierać jedną liczbę całkowitą: koszt najtańszej trasy z miasta numer k do miasta numer i. Spośród tych wierszy, wiersz o numerze k powinien zawierać liczbę 0.

Przykład

Dla danych wejściowych:

```
5 5 1 3 2
1 2
2 3
3 4
4 5
3 1
```

poprawnym wynikiem jest:

```
0
3
3
2
5
```

Wyjaśnienie do przykładu: Najtańsza trasa z miasta numer 1 do miasta numer 5 wiedzie przez miasto numer 3 lub miasto numer 4. W obu przypadkach składa się ona z jednego połączenia kolejowego i jednego lotniczego.

GOBELINY

plik źródłowy gob.*, dostępna pamięć 128 MB

W Bajtockim Muzeum Sztuk Pięknych rozpoczyna się wystawa gobelinów. Główna sala wystawowa ma, patrząc z góry, kształt wielokąta (niekoniecznie wypukłego). Na każdej ze ścian sali powieszono jeden gobelin. Każdy gobelin zajmuje dokładnie całą powierzchnię ściany.

Do sali wstawiono lampę, która ma oświetlać wystawę. Lampa świeci równomiernie we wszystkich kierunkach. Wiadomo, że niektóre z gobelinów muszą być dobrze oświetlone, a niektórych – wręcz przeciwnie – nie można wystawiać na ostre światło.

Bajtazar, kustosz muzeum, zaczął przesuwać lampę po sali, ale nie udało mu się jej ustawić tak, żeby był zadowolony. Bajtazar jest przerażony – obawia się, że będzie trzeba przewieszać gobeliny (co wymaga dużo pracy), a do otwarcia wystawy zostało bardzo niewiele czasu. Może zdołasz mu pomóc i podpowiesz, czy jego wysiłki w ogóle mają sens?

Twoim zadaniem jest rozstrzygnięcie, czy istnieje położenie lampy spełniające poniższe warunki:

- każda ściana musi być albo oświetlona w całości, albo zaciemniona w całości, w zależności od wymagań dotyczących danego gobelinu; natomiast nie może istnieć ściana, na której część pada światło, a na część nie;
- jeżeli lampa znajduje się dokładnie w linii ściany, to jej nie oświetla;
- lampy nie można wyłączyć ani zabrać z sali; musi być włączona i znajdować się wewnątrz sali (w szczególności nie może znajdować się na żadnej ze ścian ani w rogu sali).

Wejście

W pierwszym wierszu standardowego wejścia znajduje się jedna liczba całkowita t ($1 \leqslant t \leqslant 20$), oznaczająca liczbę zestawów danych. W kolejnych wierszach znajdują się opisy zestawów danych.

W pierwszym wierszu opisu znajduje się jedna liczba całkowita n ($3 \leqslant n \leqslant 1\,000$), oznaczająca liczbę ścian sali wystawowej. W kolejnych n wierszach opisany jest kształt sali. W każdym z tych wierszy znajdują się dwie liczby całkowite x_i i y_i oddzielone pojedynczym odstępem ($-30\,000 \leqslant x_i, y_i \leqslant 30\,000$ dla $i = 1, 2, \ldots, n$) – są to współrzędne narożnika sali, czyli wierzchołka wielokąta opisującego kształt sali. Wierzchołki są podane w kolejności zgodnej z kierunkiem ruchu wskazówek zegara.

W kolejnych n wierszach opisane są wymagania dotyczące gobelinów wiszących na ścianach. W każdym z tych wierszy znajduje się jedna litera `S` lub `C`, oznaczająca odpowiednio, że ściana ma być oświetlona lub zaciemniona. Litera znajdująca się w i-tym z tych wierszy (dla $1 \leqslant i \leqslant n-1$) dotyczy ściany łączącej wierzchołki i-ty oraz $(i+1)$-szy. Litera znajdująca się w ostatnim z tych wierszy dotyczy ściany łączącej ostatni wierzchołek z pierwszym.

Wielokąt opisujący kształt sali nie ma samoprzecięć, tzn. poza sąsiednimi bokami, które siłą rzeczy mają wspólny koniec, żadne dwa boki wielokąta nie mają punktów wspólnych. Żadne trzy wierzchołki wielokąta nie są współliniowe.

W testach wartych łącznie 40% punktów zachodzi dodatkowy warunek $n \leqslant 20$. Dodatkowo, w testach wartych łącznie 10% punktów wszystkie ściany mają być oświetlone.

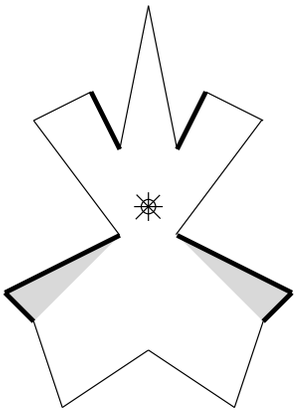
Wyjście

Dla każdego zestawu danych Twój program powinien wypisać na standardowe wyjście jeden wiersz zawierający jedno słowo: `TAK` – jeżeli da się ustawić lampę zgodnie z podanymi warunkami, lub `NIE` – w przeciwnym przypadku.

Przykład

Dla danych wejściowych:

```
1
16
5 -3
4 -4
3 -7
0 -5
-3 -7
-4 -4
-5 -3
-1 -1
-4 3
-2 4
-1 2
0 7
1 2
2 4
4 3
1 -1
C
S
S
S
S
```



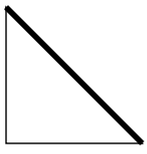
```
C
C
S
S
C
S
S
C
S
S
C
```

poprawnym wynikiem jest:

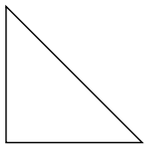
```
TAK
```

natomiast dla danych:

```
2
3
0 0
0 1
1 0
S
C
S
```



```
3
0 0
0 1
1 0
S
S
S
```



poprawnym wynikiem jest:

```
NIE
TAK
```

Wyjaśnienie do przykładu: Na rysunkach pogrubione boki oznaczają ściany, które mają być zaciemnione, a pozostałe boki – ściany, które mają być oświetlone. Rysunek do pierwszego przykładu przedstawia poprawne położenie lampy.

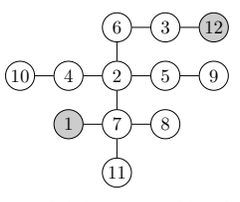
MULTIDRINK

plik źródłowy mul.*, dostępna pamięć 256 MB

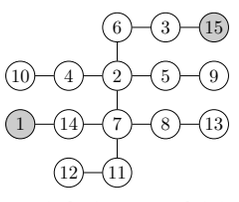
Bajtazar mieszka w Bajtowie, które słynie z tego, że przy każdym skrzyżowaniu ulic znajduje się bar mleczny. Pewnego dnia Bajtazar postanowił odwiedzić w celu tzw. „mlecznego multidrinka" wszystkie bary, każdy dokładnie raz. Bajtazar chciałby tak zaplanować trasę, żeby każdy kolejny bar był nie dalej niż dwa skrzyżowania od poprzedniego.

Skrzyżowania w Bajtowie są ponumerowane od 1 do n. Wszystkie ulice są dwukierunkowe. Między każdymi dwoma różnymi skrzyżowaniami jest tylko jedna trasa, na której żadne skrzyżowanie nie powtarza się. Bajtazar startuje przy skrzyżowaniu numer 1 i kończy przy skrzyżowaniu numer n.

Twoim zadaniem jest wyznaczenie jakiejkolwiek trasy spełniającej wymagania Bajtazara lub wypisanie informacji o braku takiej trasy.



Trasą spełniającą warunki zadania jest na przykład:
$1, 11, 8, 7, 5, 9, 2, 10, 4, 6, 3, 12$.



Nie ma żadnej trasy spełniającej warunki zadania.

Wejście

W pierwszym wierszu standardowego wejścia znajduje się jedna liczba całkowita n ($2 \leqslant n \leqslant 500\,000$) oznaczająca liczbę skrzyżowań w Bajtowie. Każdy z kolejnych $n-1$ wierszy zawiera dwie różne liczby całkowite a_i oraz b_i ($1 \leqslant a_i, b_i \leqslant n$), oddzielone pojedynczym odstępem, reprezentujące ulicę łączącą skrzyżowania o numerach a_i i b_i.

W testach wartych łącznie 65% punktów zachodzi dodatkowy warunek $n \leqslant 5\,000$.

Wyjście

Jeśli nie istnieje żadna trasa spełniająca wymagania Bajtazara, w pierwszym i jedynym wierszu standardowego wyjścia Twój program powinien wypisać jedno słowo „BRAK" (bez cudzysłowów). W przeciwnym przypadku Twój program powinien wypisać n wierszy, z których i-ty powinien zawierać numer i-tego skrzyżowania na przykładowej trasie spełniającej warunki Bajtazara. W pierwszym wierszu powinna znaleźć się liczba 1, a w n-tym wierszu – liczba n.

Przykład

Dla danych wejściowych:

```
12
1 7
7 8
7 11
7 2
2 4
4 10
2 5
5 9
2 6
3 6
3 12
```

jednym z poprawnych wyników jest:

```
1
11
8
7
5
9
2
10
4
6
3
12
```

natomiast dla danych:

```
15
1 14
14 7
7 8
7 11
7 2
2 4
4 10
2 5
5 9
2 6
3 6
3 15
11 12
8 13
```

poprawnym wynikiem jest:

```
BRAK
```

TAKSÓWKI

plik źródłowy tak.*, dostępna pamięć 64 MB

Bajtazar chce przejechać taksówką z miejscowości Bajtodziura do miejscowości Bajtodół, odległej od Bajtodziury o m km. W odległości d km od Bajtodziury na trasie między tymi miastami znajduje się baza taksówek dysponująca n taksówkami, ponumerowanymi od 1 do n. Taksówka numer i ma zapas benzyny wystarczający na przejechanie x_i km.

Bajtazar może się przesiadać, zmieniając taksówki. Wszystkie taksówki wyruszają z bazy, ale nie muszą do niej wracać. Twoim zadaniem jest sprawdzenie, czy można przewieźć Bajtazara z Bajtodziury do Bajtodołu, a jeżeli tak, to jaka jest minimalna liczba taksówek, jakie należy wykorzystać.

Wejście

W pierwszym wierszu standardowego wejścia znajdują się trzy liczby całkowite m, d oraz n ($1 \leqslant d \leqslant m \leqslant 10^{18}$, $1 \leqslant n \leqslant 500\,000$), pooddzielane pojedynczymi odstępami. Oznaczają one odpowiednio: odległość z Bajtodziury do Bajtodołu, odległość z Bajtodziury do bazy taksówek oraz liczbę taksówek znajdujących się w bazie. W drugim wierszu wejścia znajduje się n liczb całkowitych $x_1, x_2, \ldots, x_n$ ($1 \leqslant x_i \leqslant 10^{18}$), pooddzielanych pojedynczymi odstępami. Liczba x_i oznacza dystans (w km), jaki maksymalnie może przejechać taksówka numer i.

W testach wartych łącznie 40% punktów zachodzi dodatkowy warunek $n \leqslant 5\,000$.

Wyjście

Twój program powinien wypisać na standardowe wyjście jedną liczbę całkowitą: minimalną liczbę taksówek, którymi musi jechać Bajtazar, aby dostać się z Bajtodziury do Bajtodołu. Jeżeli nie jest to możliwe, Twój program powinien wypisać liczbę 0.

Przykład

Dla danych wejściowych:
```
42 23 6
20 25 14 27 30 7
```
poprawnym wynikiem jest:
```
4
```

Wyjaśnienie do przykładu: Bajtazar może jechać kolejno taksówkami numer: 4, 5, 1 i 2.

USUWANKA

plik źródłowy usu.*, dostępna pamięć 64 MB

Mała Bajtosia dostała w prezencie grę o nazwie Usuwanka. W tej grze dany jest ciąg n przylegających do siebie klocków, ponumerowanych kolejno od 1 do n. Każdy klocek jest albo biały, albo czarny, przy czym białych klocków jest k razy więcej niż czarnych. Gra jest jednoosobowa. Celem gry jest usunięcie, za pomocą określonych ruchów, wszystkich klocków z ciągu.

Pojedynczy ruch polega na usunięciu dokładnie k białych i jednego czarnego klocka bez zmiany pozycji pozostałych klocków. Ruch jest dozwolony, jeśli między żadnymi dwoma usuwanymi w tym ruchu klockami nie ma „dziury", czyli pustego miejsca po uprzednio usuniętym klocku.

Pomóż Bajtosi, ona tak bardzo Cię prosi... Znajdź dowolny ciąg dozwolonych ruchów prowadzący do usunięcia wszystkich klocków.

Wejście

W pierwszym wierszu standardowego wejścia znajdują się dwie liczby całkowite n oraz k ($2 \leqslant n \leqslant 1\,000\,000$, $1 \leqslant k \leqslant n-1$), oddzielone pojedynczym odstępem, oznaczające liczbę klocków w grze oraz liczbę białych klocków, które należy usunąć w każdym ruchu. We wszystkich testach zachodzi warunek $k+1 \mid n$.

W drugim wierszu znajduje się napis złożony z n liter `b` oraz `c`. Kolejne litery napisu określają kolor kolejnych klocków: `b` – biały, `c` – czarny. Możesz założyć, że dla danych testowych istnieje szukany ciąg ruchów prowadzący do usunięcia wszystkich klocków.

W testach wartych łącznie 40% punktów zachodzi dodatkowy warunek $n \leqslant 10\,000$.

Wyjście

Twój program powinien wypisać na standardowe wyjście $\frac{n}{k+1}$ wierszy. Kolejne wiersze powinny opisywać kolejne ruchy. Każdy z tych wierszy powinien zawierać $k+1$ liczb, podanych w kolejności rosnącej oraz rozdzielonych pojedynczymi odstępami, oznaczających numery klocków, które należy usunąć w danym ruchu.

Przykład

Dla danych wejściowych:
```
12 2
ccbcbbbbbbcb
```
jednym z poprawnych wyników jest:
```
1 8 12
2 6 7
3 4 5
9 10 11
```

Wyjaśnienie do przykładu: Niech ␣ oznacza puste miejsce po usuniętym klocku. Wykonując podane powyżej ruchy, uzyskujemy kolejno następujące układy klocków:

1	2	3	4	5	6	7	8	9	10	11	12
c	c	b	c	b	b	b	b	b	b	c	b
␣	c	b	c	b	b	b	␣	b	b	c	␣
␣	␣	b	c	b	␣	␣	␣	b	b	c	␣
␣	␣	␣	␣	␣	␣	␣	␣	b	b	c	␣
␣	␣	␣	␣	␣	␣	␣	␣	␣	␣	␣	␣

Wskazówki dla uczestników

1. Aby Twoje rozwiązanie mogło zostać właściwie ocenione, zastosuj się do ustaleń zawartych w „Zasadach organizacji zawodów" i treściach zadań.
2. W Systemie Internetowym Olimpiady sio.mimuw.edu.pl znajdują się odpowiedzi na pytania zawodników dotyczące Olimpiady. Ponieważ odpowiedzi mogą zawierać ważne informacje dotyczące toczących się zawodów, polecamy Ci regularnie zapoznawać się z ukazującymi się odpowiedziami. Dalsze pytania należy przysyłać poprzez System Internetowy Olimpiady.
3. Przestrzegaj dokładnie warunków określonych w treści zadania.
4. Należy założyć, że dane testowe są bezbłędne, zgodne z warunkami zadania i podaną specyfikacją wejścia. Twój program nie musi tego sprawdzać.
5. Nie przyjmuj żadnych założeń, które nie wynikają z treści zadania.
6. Staraj się dobrać taką metodę rozwiązania zadania, która jest nie tylko poprawna, ale daje wyniki w jak najkrótszym czasie.
7. Ocena za rozwiązanie zadania jest określana na podstawie wyników testowania programu i uwzględnia poprawność oraz efektywność metody rozwiązania użytej w programie. W szczególności programy poprawne, lecz działające zbyt długo – zwłaszcza dla dużych rozmiarów danych – mogą zostać ocenione nisko.
8. Koniecznie zapoznaj się z materiałami dostępnymi w witrynie Olimpiady: www.oi.edu.pl.
9. **Nie udostępniaj innym swoich rozwiązań zadań przed upływem godziny zakończenia zawodów. Chroń je przed niepowołanym dostępem. Jeśli ktoś wyśle skradzione Tobie lub udostępnione przez Ciebie rozwiązania zadań, możesz zostać zdyskwalifikowany.**