MALACHOVIACUS
INFORMATICUS

Płock, 11 marca 2017 r.

Instrukcje dla zawodników

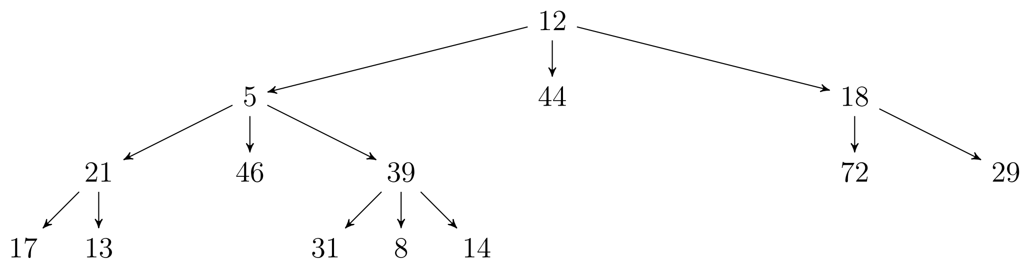
Arkusze otwieramy na wyraźne polecenie komisji. Wszystkie poniższe instrukcje zostaną odczytane i wyjaśnione.

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. Nieczytelne odpowiedzi **nie będą oceniane**.
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 **120 minut**.

Powodzenia!

ZADANIE 1
Sortujące drzewa

Drzewa to grafy, w których nie ma żadnych cykli, a każdy wierzchołek posiada co najwyżej jedną krawędź wchodzącą. Dodatkowo, do każdego wierzchołka drzewa przypisywać będziemy jakąś wartość. Wartości mogą być zupełnie dowolne, jednak w tym zadaniu ograniczymy się jedynie do liczb naturalnych.

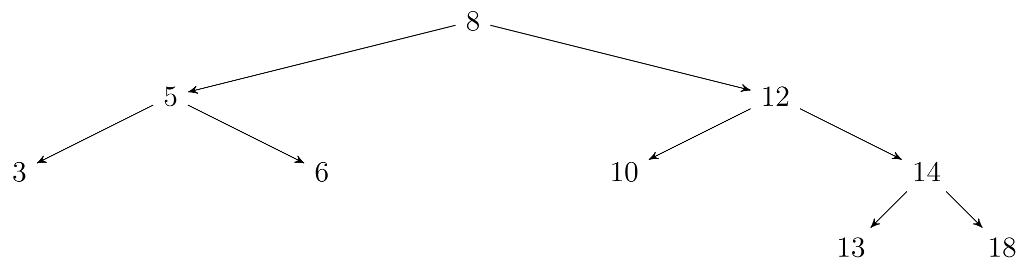


Rysunek 1: Przykładowe drzewo.

Każde drzewo posiada wierzchołek nadrzędny, zwany **korzeniem**, który jako jedyny nie posiada żadnych krawędzi wchodzących. W przykładowym drzewie powyżej jest to wierzchołek z liczbą 12.

Dzięki temu możemy w drzewie określić jasną hierarchię wśród wierzchołków. Jeżeli krawędź prowadzi z wierzchołka A do B, to mówimy, że A jest **rodzicem** B, natomiast B jest **dzieckiem** A. Przykładowo na rysunku 1 wierzchołek z liczbą 21 jest rodzicem wierzchołka z liczbą 13.

Wśród drzew wyróżniamy **drzewa binarne**, to jest takie, w których każdy wierzchołek posiada co najwyżej 2 dzieci (to znaczy 0, 1 lub 2). Możemy wtedy mówić o dziecku **lewym** i **prawym**.



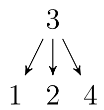
Rysunek 2: Przykładowe drzewo przeszukiwań binarnych.

Z kolei specyficznym rodzajem drzew binarnych są **drzewa przeszukiwań binarnych**, w których każdy wierzchołek posiada charakterystyczną własność. Wszystkie wierzchołki po lewej stronie dowolnego wierzchołka (rodzica): lewe dziecko, dzieci lewego dziecka, itp. mają wartości mniejsze od danego wierzchołka, natomiast wszystkie wierzchołki po prawej stronie: prawe dziecko, dzieci prawego dziecka itp. mają wartości większe.

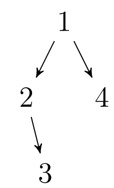
Zwróć uwagę, że w przykładowym drzewie przeszukiwań binarnych wierzchołek z liczbą 13 nie może mieć żadnych dzieci. Jego lewe dziecko musiałoby mieć wartość jednocześnie mniejszą od 13 i większą od 12. Natomiast dziecko po prawej stronie musiałoby mieć wartość jednocześnie większą od 13 i mniejszą od 14. Wśród liczb naturalnych nie da się znaleźć wartości spełniających te warunki.

PYTANIA

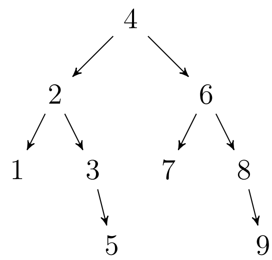
PYTANIE 1 Dlaczego poniższe drzewa nie są drzewami przeszukiwań binarnych? Wypisz wszystkie powody które możesz znaleźć.



(a) [1 punkt]

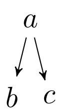


(b) [1 punkt]

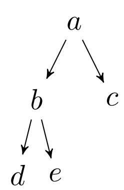


(c) [2 punkty]

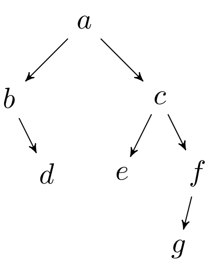
PYTANIE 2 Drzewa poniżej są drzewami przeszukiwań binarnych. Dla każdego z nich podaj liczbę najmniejszą, największą oraz środkową (t.j. taką, że w drzewie jest tyle samo liczb od niej mniejszych co liczb od niej większych).



(a) [3 punkty]



(b) [3 punkty]



(c) [3 punkty]

PYTANIE 3 W jaki sposób możemy wypisać liczby znajdujące się w drzewie przeszukiwań binarnych od najmniejszej do największej, jeżeli nie znamy żadnej z wartości znajdującej się w drzewie? Opisz w postaci listy kroków i krótko uzasadnij dlaczego twój sposób działa. [5 punktów]

Co stanie się, jeżeli wykonamy te kroki dla drzewa binarnego, które nie jest drzewem przeszukiwań binarnych? [2 punkty]

ZADANIE 2

Odczytaj, przetwórz, zapisz

Pamięć komputera składa się z ogromnej liczby **komórek**. Każda z nich jest w stanie przechować jedną liczbę naturalną (dla uproszczenia przyjmijmy, że dowolnie dużą). Komórki numerowane są kolejno od 0 – numer ten nazywamy **adresem komórki**.

Program wykonywany przez **procesor** jest listą kolejno wykonywanych **instrukcji**, z których każda wykonuje jedną prostą operację, jak dodawanie czy mnożenie. Każda z nich posiada adres komórki wyjściowej, w której przechowany będzie wynik operacji, oraz co najwyżej dwie wartości wejściowe: adresy komórek lub liczby. W tym zadaniu będziemy używać następujących instrukcji:

- `a <- b + c`, która dodaje wartości b i c, po czym zapisuje wynik w a.
- `a <- b * c`, która mnoży wartości b i c, po czym zapisuje wynik w a.

Po lewej stronie strzałki zawsze znajdować będzie się adres komórki, do której zapisujemy wynik, natomiast po prawej stronie może znajdować się adres (zapisywany `@N`, gdzie `N` to dowolna liczba naturalna) lub liczba (zapisywana `#N`). Przykładowo:

- `1 <- @2 + #3` doda 3 do wartości komórki 2, wynik zapisze w komórce 1,
- `1 <- @2 * @3` pomnoży wartości komórek 2 i 3, wynik zapisze w komórce 1.

Każda instrukcja wykonywana jest w 3 etapach: **odczyt**, **przetwarzanie** i **zapis**. W pierwszym etapie procesor odczytuje wartości odpowiednich komórek z pamięci. W drugim wykonywana jest właściwa operacja, a w trzecim wynik operacji zapisywany jest w pamięci. Stąd wykonanie instrukcji `0 <- @0 + #1` zwiększy wartość w komórce 0 o 1: najpierw wczytana zostanie obecna wartość komórki 0, potem dodamy do niej 1, a na końcu nowa wartość zostanie zapisana w pamięci.

Wynikiem działania programu jest zawsze wartość znajdująca się w komórce 0 po wykonaniu ostatniej instrukcji. Początkowa wartość w komórce 0 jest równa 0, pozostałe możemy ustalić dowolnie.

Przykładowo poniższy program oblicza sumę wartości z komórek 1, 2 i 3:

```
0 <- @1 + @2
0 <- @0 + @3
```

PYTANIA

PYTANIE 1 Jeżeli początkowa wartość komórki 1 to 3, a komórki 2 to 4, jakie wartości będą znajdować się w komórkach 0, 1 i 2 przed i po wykonaniu każdej z instrukcji poniższego programu? Jaki będzie wynik działania tego programu? [2 punkty]

```
1 <- @1 * @1
2 <- @2 * @2
0 <- @1 + @2
```

Jaki będzie wynik działania tego programu, jeżeli początkowa wartość komórki 1 to a, a komórki 2 to b? [1 punkt]

Pytanie 2 Napisz program, który obliczy wartość wyrażenia $(n+1)^2$, gdzie n to początkowa wartość w komórce 1. [3 punkty]

Pytanie 3 Bajtek próbuje napisać pewien program. Potrzebna jest mu instrukcja `a <- b`, która umieści wartość b w komórce a. Niestety, używany przez niego procesor potrafi jedynie dodawać i mnożyć. Jak rozwiązać ten problem bez wprowadzania zmian do procesora? Pamiętaj, że b może być zarówno liczbą jak i adresem. [3 punkty]

Pytanie 4 Każdy z etapów wykonania instrukcji przez procesor zajmuje jedną jednostkę czasu, więc wykonanie jednej instrukcji zajmuje 3 jednostki.

Jako że za każdy z etapów odpowiedzialna jest inna część procesora, mogłyby być one wykonywane jednocześnie – ale nie dla tej samej operacji. Bajtek zaproponował, żeby odczyt dla kolejnej operacji wykonywać jednocześnie z przetwarzaniem obecnej operacji i zapisywaniem wyniku poprzedniej.

Czas	Odczyt	Przetwarzanie	Zapis
	Instrukcja 1	*nic*	*nic*
	Instrukcja 2	Instrukcja 1	*nic*
	Instrukcja 3	Instrukcja 2	Instrukcja 1
	nic	Instrukcja 3	Instrukcja 2
	nic	*nic*	Instrukcja 3

(a) Ile czasu byłoby potrzebne na wykonanie jednej instrukcji? A na wykonanie programu składającego się z 5 instrukcji? [2 punkty]

(b) Bajtek obawia się, że taka zmiana może wpłynąć na wynik działania niektórych programów. Napisz program, którego wynik zmieni się po wprowadzeniu ulepszenia zaproponowanego przez Bajtka. Jaki będzie wynik działania tego programu w obu przypadkach? [4 punkty]

(c) Napisz program, który po wprowadzeniu ulepszenia Bajtka szybciej zakończy działanie, a jego wynik nie zmieni się. [3 punkty]

(d) Jak można zapobiec zmianie wyniku programu po wprwadzeniu ulepszenia Bajtka? [2 punkty]

Zadanie 3

Funkcje rekurencyjne

Funkcja to matematyczne określenie przypisania elementów jednego zbioru elementom innego zbioru, na przykład jednej liczbie naturalnej – innej liczby naturalnej.

Funkcje możemy przedstawiać na różne sposoby: słownie, używając grafu, tabelki, wzoru lub wykresu. Możemy też opisywać je **rekurencyjnie**.

$S(n)$ to suma liczb naturalnych od 1 do n włącznie.

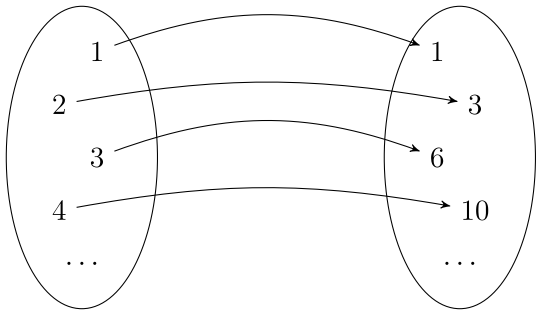
(a) słownie

$$S(n) = 1 + 2 + \ldots + n$$
$$S(n) = \frac{n(n+1)}{2}$$

(b) wzór

n	1	2	3	4	$\cdots$
$f(n)$	1	3	6	10	$\cdots$

(c) tabelka



(d) graf

Rysunek 5: Różne sposoby przedstawienia funkcji S

Rekurencyjny opis funkcji to taki, który wykorzystuje sam siebie. Na przykład sumę liczb naturalnych od 1 do n włącznie możemy obliczyć jako sumę liczb naturalnych od 1 do $n - 1$ powiększoną o n. Używając wzoru:

$$S(n) = n + S(n-1)$$

Taki opis funkcji nie jest jednak kompletny. Gdybyśmy spróbowali, na przykład, obliczyć $S(3)$, nie otrzymamy wyniku. Musielibyśmy rozwinąć wzór na S nieskończenie wiele razy:

$$\begin{aligned} S(3) \quad &= 3 + S(2) \\ &= 3 + 2 + S(1) \\ &= 3 + 2 + 1 + S(0) \\ &= 3 + 2 + 1 + 0 + S(-1) \\ &= 3 + 2 + 1 + 0 + (-1) + S(-2) \\ &\vdots \end{aligned}$$

Musimy więc do naszej definicji dodać jeszcze jedną regułę, **podstawę rekurencji**, to znaczy pewną wartość funkcji, która nie będzie zdefiniowana rekurencyjnie, na przykład:

$$S(1) = 1$$

Wtedy nasze obliczenie dla $S(3)$ wyglądać będzie następująco:

$$\begin{aligned} S(3) &= 3 + S(2) \\ &= 3 + 2 + S(1) \\ &= 3 + 2 + 1 = \underline{6} \end{aligned}$$

Zazwyczaj obie części takiego rekurencyjnego opisu funkcji zapisujemy razem, łącząc je klamerką i dodając warunki:

$$S(n) = \begin{cases} 0 & \text{gdy } n = 0 \\ n + S(n-1) & \text{gdy } n > 0 \end{cases}$$

Pytania

PYTANIE 1 Dana jest funkcja

$$f(n) = \begin{cases} 1 & \text{gdy } n = 0 \\ 2 \cdot f(n-1) & \text{gdy } n > 0 \end{cases}$$

Narysuj tabelkę z wartościami tej funkcji dla $n = 0, 1, 2, 3$. [2 punkty]

Opisz $f(n)$ słownie lub wzorem, bez użycia rekurencji. [2 punkty]

PYTANIE 2 Dana jest funkcja

$$g(n) = \begin{cases} n & \text{gdy } n \text{ jest nieparzyste} \\ g\left(\frac{n}{2}\right) & \text{gdy } n \text{ jest parzyste} \end{cases}$$

Oblicz wartość $g(28)$. [3 punkty]

PYTANIE 3 Przedstaw poniższą funkcję rekurencyjnie. [4 punkty]

$$S_k(n) = 1^2 + 2^2 + 3^2 + \ldots + n^2$$

PYTANIE 4 *Silnia* z n (zapisywane $n!$) to iloczyn kolejnych liczb naturalnych od 1 do n włącznie, np:

- $1! = 1$
- $2! = 1 \cdot 2 = 2$
- $5! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 = 120$

Przedstaw $n!$ rekurencyjnie. [4 punkty]

PYTANIE 5 Zapisz $(n+1)^2$ w postaci sumy algebraicznej. Na tej podstawie przedstaw funkcję $f(n) = n^2$ rekurencyjnie, wykorzystując jedynie dodawanie i odejmowanie. [5 punktów]