

Algorytmy i Struktury Danych
egzamin na studia uzupełniające
wrzesień 2013 r.

Zadanie 1 (32 punktów)

Dany jest nieuporządkowany ciąg n -elementowy i liczba naturalna k , gdzie $0 < k \leq n$. Udowodnij, że w modelu drzew decyzyjnych wyznaczenie k -tego co do wielkości elementu w nieuporządkowanym ciągu n -elementowym oraz wydzielenie z tego ciągu $k-1$ elementów nie większych i $n-k$ elementów nie mniejszych od pivotu wymaga wykonania $\Omega(\log k + \log \binom{n}{k})$ porównań. Uzasadnij swoje obliczenia.

Uwaga. Opisz pojęcie drzewa decyzyjnego.

Zadanie 2 (36 punkty)

Wyjaśnij, na czym polega adresowanie otwarte w tablicach z haszowaniem. Opisz dokładniej jakiś sposób rozwiązywania konfliktów w takiej tablicy, które mogą się pojawić podczas wstawiania do niej nowych elementów. Opisz, jak zaimplementować usuwanie niepotrzebnych elementów z tablicy z haszowaniem.

Zadanie 3 (32 punkty)

Dany jest n -elementowy rosnący ciąg liczb całkowitych $A = (a_0, a_1, \dots, a_{n-1})$, taki że $a_0 < a_1 < \dots < a_{n-1}$. Można założyć, że liczby te są zapisane w tablicy $A[0 \dots n-1]$. Opisz efektywny algorytm (działający poniżej czasu liniowego), który sprawdzi czy w tym ciągu znajduje się chociaż jeden taki element, że na i -tej pozycji znajduje się wartość i , czyli $A[i] = i$ dla pewnego $i \in \{0 \dots n-1\}$. Jeśli taki element istnieje, to należy wypisać jedną taką wartość, a jeśli nie ma takiego elementu, to należy wypisać -1 . Uzasadnij poprawność twojego algorytmu i przeanalizuj jego złożoność obliczeniową.

Przykład. Dla ciągu liczb $[-3, -1, 1, 3, 4, 5, 7, 9]$ algorytm może odpowiedzieć liczbą 4 (albo 3 lub 5), a dla ciągu liczb $[-1, 2, 5, 7, 8, 9]$ algorytm ma odpowiedzieć wartością -1 .

Algorytmy i Struktury Danych

egzamin na studia uzupełniające

luty 2013r.

Zadanie 1 (36 punktów)

Jakie jest dolne ograniczenie na liczbę porównań w *problemie sortowania*? Zakładamy, że sortujemy tylko w oparciu o porównywanie kluczy.

1. Oszacuj asymptotycznie od dołu liczbę porównań jaką musi wykonać w najgorszym przypadku algorytm sortujący dane rozmiaru n ; wykorzystaj do tego celu drzewa decyzyjne.
2. W oparciu o przedstawioną granicę dolną wylicz, ile porównań musi wykonać w najgorszym przypadku algorytm sortujący 5 elementów; następnie skonstruuj taki algorytm (sortujący optymalnie) i zapisz go w pseudokodzie.
3. Wybierz i opisz jeden asymptotycznie optymalny co do liczby porównań algorytm sortujący; opisz krótko ideę jego działania; przeanalizuj jego złożoność czasową.

Zadanie 2 (32 punkty)

Dana jest lista n wykładów, które mają się odbywać w tej samej sali. Każdy z wykładowców upiera się przy swoim terminie przeprowadzenia zajęć: $[p_1, k_1), [p_2, k_2), \dots, [p_n, k_n)$. Nie można wszystkich usatysfakcjonować, gdyż dysponujemy tylko jedną salą wykładową.

1. Podaj algorytm, który wybierze wykłady w taki sposób, aby odbyło się ich jak najwięcej.
2. Koniecznie uzasadnij poprawność Twojego rozwiązania (dowód nie wprost).
3. Czy algorytm ten będzie skuteczny, gdybyśmy chcieli zmaksymalizować czas wykorzystania sali?

Zadanie 3 (32 punkty)

Zaprojektuj strukturę danych umożliwiającą efektywne wykonywanie (najlepiej w czasie logarytmicznym względem bieżącej długości ciągu) następujących operacji na początkowo pustym ciągu liczbowym S :

- $S.insert(x, j)$: wstawienie liczby x na j -tą pozycję do ciągu S ;
- $S.sum(j)$: obliczenie sumy j początkowych elementów w ciągu S ($\sum_{i=0}^{j-1} s_i$).

W powyższych operacjach parametr j spełnia warunek $0 \leq j \leq |S|$.

1. Opisz ideę rozwiązania tego problemu. Zaprojektuj strukturę umożliwiającą efektywne wykonywanie na niej wymienionych operacji, wykorzystując jakąś znaną strukturę danych realizującą efektywnie operację *insert*.
2. Opisz dokładnie modyfikacje wprowadzone do tej struktury i do procedury *insert*; uzasadnij, że nie zmienia to poprawnego i efektywnego działania tej procedury.

3. Procedurę *sum* zapisz w pseudokodzie, opisz jej działanie i przeanalizuj złożoność obliczeniową (czas i pamięć).

Uwaga. Ciąg S jest indeksowany od zera, czyli $S = (s_0, s_1, \dots, s_{n-1})$, dla $|S| = n$.

Przykład. Przy wstawianiu nowej liczby na j -tą pozycję, wszystkie liczby od pozycji j -tej włącznie są przesuwane o jedną pozycję dalej. Niech $S = (a, b, c)$ będzie 3-elementowym ciągiem liczbowym. Wówczas:

- wykonując operację $S.insert(x, 1)$, czyli wstawiając liczbę x na pozycję 1, otrzymamy ciąg (a, x, b, c) ;
- wykonując operację $S.insert(y, 0)$, wstawimy y na początek ciągu otrzymując (y, a, b, c) ;
- a wykonując operację $S.insert(z, 3)$, wstawimy z na koniec ciągu otrzymując (a, b, c, z) .

Algorytmy i Struktury Danych

egzamin na magisterskie studia uzupełniające

wrzesień 2014 r.

Zadanie 1: dolna granica dla problemu skalania (24 punkty)

Rozważmy kolejkę priorytetową Q z operacjami:

- $Q.create(S)$ — utworzenie kolejki priorytetowej złożonej z elementów zbioru S ,
- $Q.insert(x)$ — wstawienie nowego elementu x do kolejki,
- $Q.top()$ — sprawdzenie, który element w kolejce ma najwyższy priorytet,
- $Q.extract-max()$ — wyciągnięcie z kolejki elementu o najwyższym prioryecie.

Wyjaśnij, dlaczego złożoność czasowa operacji $extract-max$ wynosi $\Omega(\log n)$, gdzie n jest liczbą wszystkich elementów w kolejce priorytetowej.

Zadanie 2: graf dynamiczny (44 punktów)

Graf dynamiczny to graf, w którym można dodawać i usuwać zarówno krawędzie jak i wierzchołki. Wierzchołki niech będą oznaczone unikatowymi liczbami naturalnymi; krawędzie to po prostu pary wierzchołków.

Zaprojektuj strukturę danych dla grafu dynamicznego, która pozwoli efektywnie realizować następujące operacje:

- sprawdzenie, czy wierzchołek o podanym numerze istnieje w grafie;
- dodanie nowego wierzchołka do grafu;
- usunięcie zadanego wierzchołka z grafu wraz ze wszystkimi incydentnymi krawędziami;
- wyznaczenie stopnia zadanego wierzchołka;
- sprawdzenie, czy podana krawędź istnieje w grafie;
- dodanie nowej krawędzi do grafu;
- usunięcie zadanej krawędzi z grafu.

Opisz krótko działanie wymienionych operacji na Twojej strukturze. Jakie będą złożoności czasowe tych operacji.

Zadanie 3: optymalizacja drzewa AVL (32 punkty)

Drzewo AVL jest zrównoważonym drzewem BST. Tradycyjna implementacja tego drzewa jest taka, że w każdym węźle tej struktury pamiętamy klucz i różnicę wysokości poddrzew doczepionych do tego węzła (plus oczywiście wskaźniki do poddrzew). Chcielibyśmy jednak pozbyć się tej dodatkowej informacji o różnicy wysokości poddrzew w każdym węźle.

- Krótko ale precyzyjnie opisz strukturę drzewa AVL.
- Wyjaśnij, w jaki sposób można się pozbyć informacji o różnicy wysokości poddrzew z każdego węzła.
- Uzasadnij, że zaproponowana modyfikacja nie spowoduje asymptotycznego pogorszenia się czasu działania żadnej operacji słownikowej w takim drzewie.

Algorytmy i Struktury Danych

egzamin na studia uzupełniające

luty 2014r.

Zadanie 1: dolna granica dla problemu skalania (32 punkty)

Dane są dwa posortowane n -elementowe ciągi. Elementy obu ciągów należą do pewnego zbioru z porządkiem liniowym. Udowodnij, że w modelu drzew decyzyjnych dolne ograniczenie na liczbę porównań dla *problemu skalania* takich ciągów wynosi $\Omega(n)$.

Zadanie 2: duża nieainicjalizowana tablica (36 punktów)

Zaprojektuj strukturę danych, która umożliwia poprawne wykonywanie następujących operacji tablicowych na dużej *niezainicjalizowanej tablicy*:

- $create(n)$ — utworzenie pustej tablicy o n elementach, gdzie n jest zadaną liczbą naturalną; tablica jest indeksowana od 0 do $n - 1$; początkowo wszystkie sloty w tablicy są puste;
- $empty(i)$ — sprawdzenie czy i -ty slot w tablicy jest pusty;
- $clear(i)$ — usunięcie ewentualnej wartości z i -tego slotu w tablicy;
- $size()$ — podanie liczby wszystkich wykorzystanych slotów w tablicy;
- $read(i)$ — odczytanie wartości z i -tego slotu w tablicy (indeks i powinien być $0 \leq i < n$);
- $write(i, x)$ — wpisanie wartości x do i -tego slotu w tablicy (indeks i powinien być $0 \leq i < n$).

W każdym momencie działania programu struktura powinna zajmować $O(n)$ komórek pamięci. Opisz, w jaki sposób zaimplementowane powinny być poszczególne instrukcje. Jaki jest czas ich wykonania? Pamiętaj, że powinien on być atrakcyjnie mały nawet w najgorszym przypadku.

Zadanie 3: najdłuższy podciąg palindromiczny (32 punkty)

Ciąg nazywamy *palindromicznym* jeśli wygląda tak samo czytany od początku do końca i od końca do początku. Opracuj algorytm obliczania długości najdłuższego podciągu palindromicznego dla podanego na wejściu n -elementowego ciągu $X = (x_1, \dots, x_n)$. Uzasadnij poprawność opisanego algorytmu. Jaka jest jego złożoność obliczeniowa (czasowa i pamięciowa)?

Algorytmy i Struktury Danych

egzamin na magisterskie studia uzupełniające

18 lutego 2015 r.

Zadanie 1: problemy P i NP (20 punktów)

Wyjaśnij, co to jest problem decyzyjny. Jak są zdefiniowane klasy problemów P i NP ? Podaj przykład jednego problemu należącego do P i jednego problemu należącego do NP . Jaka jest zasadnicza różnica między tymi klasami problemów? Czy te klasy problemów są takie same, a więc czy $P = NP$?

Zadanie 2: pętla w liście jednokierunkowej (44 punktów)

Rozważmy listę jednokierunkową, o której nie wiemy, czy się ona zapętla czy nie. Zapętlenie w liście polega na tym, że w ostatnim węźle wskaźnik na następnika nie jest pusty tylko wskazuje na jakiś węzeł wewnątrz listy. Opracuj i opisz algorytm, który sprawdzi, czy lista jest zapętłona czy nie w liniowym czasie $O(n)$, gdzie n to liczba wszystkich węzłów w liście.

Prostsza wersja tego zadania dotyczy sytuacji, gdy w ostatnim węźle wskaźnik na następnika jest albo pusty albo wskazuje na pierwszy element.

Zadanie 3: szybkie przesiewanie w dół w drzewie dwumianowym (28 punktów)

Dane jest drzewo dwumianowe stopnia k , które zawiera 2^k węzłów. W drzewie tym jest zachowany porządek kopcowy. Następnie zmieniamy w dowolnym wybranym węźle pamiętaną w nim wartość. Porządek kopcowy może więc zostać zaburzony. Jeśli zmienimy wartość na większą, to zwykła procedura przesiewania w górę przywróci ten porządek w $O(\log n)$ krokach, a więc w akceptowalnym dla nas czasie. Jeśli natomiast zmienimy wartość na mniejszą, to przywrócenie porządku kopcowego poprzez proste przesiewanie w dół będzie działać istotnie dłużej, bo w $O(\log^2 n)$ krokach. Opracuj metodę, która po zmianie wartości na mniejszą we wskazanym węźle drzewa dwumianowego przywróci porządek kopcowy w czasie $O(\log n)$.

Najpierw napisz definicję drzewa dwumianowego, następnie na czym polega porządek kopcowy w drzewie, potem przedstaw ideę rozwiązania problemu (może to być pseudokod z obszernymi komentarzami) a na koniec uzasadnij, że czas działania opisanego algorytmu rzeczywiście jest logarytmiczny.

Egzamin na studia II stopnia

19 września 2013 r.

Zadania z analizy numerycznej

1. (25 punktów) Niech L_n będzie wielomianem interpolującym funkcję $f(x) = \exp x$ w zerach $(n+1)$ -szego wielomianu Czebyszewa. Jaka wartość n gwarantuje, że zachodzi nierówność

$$\max_{-1 \leq x \leq 1} |f(x) - L_n(x)| \leq 10^{-5}?$$

2. (25 punktów) Odwrotność liczby c można obliczać bez wykonywania dzielenia, za pomocą wzoru

$$x_{n+1} := x_n(2 - cx_n) \quad (n = 0, 1, \dots).$$

Uzasadnić ten fakt.

3. (25 punktów) Sprawdzić, czy funkcja

$$s(x) := \begin{cases} (x+1)[1+(x+1)^2], & x \in [-1, 0], \\ 4+(x-1)[1+(x-1)^2], & x \in [0, 1] \end{cases}$$

jest naturalną funkcją sklejaną III stopnia, interpolującą funkcję f , o wartościach

$$f(-1) = 0, \quad f(0) = 2, \quad f(1) = 4.$$

4. (25 punktów) Ile co najmniej podprzedziałów trzeba uwzględnić w złożonym wzorze trapezów, aby obliczyć całkę

$$\int_1^2 (x + e^{-x^2}) dx$$

z błędem mniejszym od $5 \cdot 10^{-8}$?

Egzamin na studia II stopnia

15 lutego 2013 r.

Zadania z analizy numerycznej

1. (25 punktów) Funkcja f o wartościach

x	-2	-1	0	1	2	3
$f(x)$	1	4	11	16	13	-4

jest prawdopodobnie wielomianem trzeciego stopnia. Jak można to sprawdzić?

2. (25 punktów)

Czy istnieją takie stałe a , b , c i d , że funkcja

$$s(x) = \begin{cases} ax^3 + x^2 + cx & (-1 \leq x \leq 0), \\ bx^3 + x^2 + dx & (0 \leq x \leq 1) \end{cases}$$

jest naturalną funkcją sklejaną III stopnia, przyjmującą w punktach -1 , 0 , 1 te same wartości, co funkcja $f(x) = |x|$?

3. (25 punktów) Wykazać, że wielomian $w(x) = 0$ jest drugim wielomianem optymalnym dla funkcji $f(x) = \sin 4x$ w sensie aproksymacji jednostajnej w przedziale $[0, 2\pi]$.

4. (25 punktów) Obliczyć element T_{30} następującej tablicy Romberga przybliżeń pewnej całki:

$$\begin{array}{ccccccc} T_{00} = 4092 & & & & & & \\ T_{01} & & T_{10} & & & & \\ T_{02} = 312 & & T_{11} = 12 & & T_{20} & & \\ T_{03} & & T_{12} & & T_{21} = -4 & & T_{30} = \boxed{} \end{array}$$

**Egzamin na studia uzupełniające II stopnia
w roku akademickim 2013/2014**

Zadania z analizy numerycznej

Termin: 19 września

1. a) 10 punktów Sprawdź czy podany niżej algorytm obliczania wartości wyrażenia $\frac{a(d+1)}{b+c+b\bar{d}}$ jest algorytmem numerycznie poprawny.
- S:=d+1;
S:=c/S;
S:=b+S;
S:=a/S;

return(S) .
- b) 10 punktów Do rozwiązania zadania obliczeniowego $\mathcal{A}$ użyto algorytmu numerycznie poprawnego. Czy można mieć pewność, że otrzymany w ten sposób wynik jest bliski rzeczywistego rozwiązania zadania $\mathcal{A}$? Odpowiedź uzasadnij.
2. 40 punktów Niech dana będzie funkcja ciągła $f : \mathbb{R} \rightarrow \mathbb{R}$, o której wiadomo, że w przedziale $[a, b]$ ma dokładnie jedno miejsce zerowe α . Zaproponuj metodę wyznaczania takiego przybliżenia $z \in [a, b]$ miejsca zerowego α , że $|\alpha - z| < \varepsilon$, gdzie $\varepsilon > 0$ jest dane.
3. 40 punktów Opisz szczegółowo metodę Romberga obliczania przybliżonej wartości całki $\int_a^b f(x) dx$. Następnie zaproponuj jej efektywną implementację, która będzie minimalizowała liczbę wywołań procedury wyznaczającej dla danego argumentu wartość funkcji podcałkowej.

**Egzamin na studia uzupełniające II stopnia
w roku akademickim 2012/2013**

Zadania z analizy numerycznej

Termin: 19 lutego

1. a) 15 punktów Wytlumacz na czym polega i kiedy występuje zjawisko *utraty cyfr znaczących* wyniku.
- b) 15 punktów Określ dla jakich $x \in \mathbb{R}$ obliczanie wartości wyrażenia

$$\frac{1}{\sqrt{x^2 + 2} + x}$$

może wiązać się z utratą cyfr znaczących wyniku. Zaproponuj sposób obliczenia wyniku dokładniejszego.

2. 40 punktów Załóżmy, że dysponujemy procedurą `Integral(h)` obliczającą z dokładnością bliską maszynowej wartość całki

$$\int_a^b h(x) dx$$

dla dowolnej funkcji ciągłej $h : [a, b] \rightarrow \mathbb{R}$. Opisz w szczegółach algorytm numerycznego wyznaczania wielomianu w_n o najmniejszym możliwym stopniu $n \in \mathbb{N}$, który spełnia warunek

$$\int_a^b \left(f(x) - w_n(x) \right)^2 dx < \varepsilon,$$

gdzie f jest daną funkcją ciągłą w przedziale $[a, b]$, natomiast $\varepsilon > 0$ jest ustalone.

3. 40 punktów Niech dana będzie macierz nieosobliwa $A \in \mathbb{R}^{n \times n}$. Zaproponuj algorytm obliczania macierzy A^{-1} oraz określ jego złożoność.

Egzamin na studia uzupełniające II stopnia
w roku akademickim 2014/2015

Zadania z analizy numerycznej

Termin: 18 lutego

1. a) 10 punktów Podaj definicję ciągu wielomianów ortogonalnych w przedziale $[a, b]$ z wagą p .
b) 10 punktów Przypomnij i krótko opisz dwa zastosowania wielomianów ortogonalnych w analizie numerycznej.
2. 40 punktów Zaproponuj algorytm obliczania wyznacznika macierzy $A \in \mathbb{R}^{n \times n}$. Określ jego złożoność.
3. 40 punktów Niech będzie $\mathbf{x} := [x_0, x_1, \dots, x_n]$ ($x_0 < x_1 < \dots < x_n$), $\mathbf{y} := [y_0, y_1, \dots, y_n]$ oraz $\mathbf{z} := [z_0, z_1, \dots, z_m]$. Niech s_n oznacza naturalną funkcję sklejaną trzeciego stopnia (w skrócie: NFS3) spełniającą warunki $s_n(x_k) = y_k$ ($0 \leq k \leq n$). W języku PWO++ procedura `NSpline3(x,y,z)` wyznacza wektor $\mathbf{Z} := [s_n(z_0), s_n(z_1), \dots, s_n(z_m)]$, z tym, że **musi być** $m < 2n$. Załóżmy, że wartości pewnej funkcji ciągłej f znane są **jedynie** w punktach $x_0 < x_1 < \dots < x_{100}$. Wiadomo, że NFS3 odpowiadająca danym $(x_k, f(x_k))$ ($0 \leq k \leq 100$) bardzo dobrze przybliża funkcję f . Wywołując procedurę `NSpline3` **tylko raz**, opracuj algorytm numerycznego wyznaczania przybliżonych wartości wszystkich miejsc w przedziale $[x_0, x_{100}]$, w których funkcja f ma ekstrema lokalne.

Matematyka dyskretna

egzamin na studia II stopnia w 2013 r.

Zadanie 1 (50 punktów)

Niech w_n oznacza liczbę ciągów złożonych z n liter należących od 26 literowego alfabetu, w których litery a i b nie występują obok siebie. Znajdź zwarty wzór na w_n .

Zadanie 2 (50 punktów)

Ucieczka z labiryntu. Dany jest dowolny spójny graf G . Pokaż że następująca metoda zapewnia przejście wszystkich jego krawędzi zaczynając od wierzchołka v :

1. nigdy nie przechodzimy tej samej krawędzi dwa razy w tym samym kierunku;
2. jeśli napotkamy wierzchołek $x \neq v$, w którym dotychczas nie byliśmy, to zaznaczamy krawędź którą do niego dotarliśmy;
3. używamy tej zaznaczonej krawędzi do opuszczenia x dopiero gdy nie ma innej alternatywy.

Pokaż, że po przejściu całego G wrócimy do wierzchołka v .

Matematyka dyskretna

egzamin na studia II stopnia w 2013 r.

Zadanie 1 (50 punktów)

Ile jest permutacji 26 liter alfabetu łacińskiego, które **nie zawierają** żadnego ze słów: *fish, rat, bird*.

Zadanie 2 (50 punktów)

Niech G będzie n -wierzchołkowym grafem prostym, a P najkrótszą drogą łączącą pewne dwa jego wierzchołki. Pokaż, że

$$\sum_{v \in P} \deg(v) \leq 3|V(G)|.$$

Matematyka dyskretna

egzamin na studia II stopnia w 2013 r.

Zadanie 1 (50 punktów)

Ile jest sposobów przedstawienia n jako sumy s składników całkowitych dodatnich, z których każdy jest nie większy niż k ?

Zadanie 2 (50 punktów)

Niech $g(G)$ oznacza długość najkrótszego cyklu w grafie G . Udowodnij, że jeśli minimalny stopień wierzchołka G jest co najmniej 3, to $g(G) \leq 2 \log n$.

Matematyka dyskretna

egzamin na studia II stopnia w 2015 r.

Zadanie 1 (50 punktów)

Wyznacz zwarty wzór na a_n , gdzie

$$a_n = \sum_{k=1}^n k(k-1)2^k.$$

Zadanie 2 (50 punktów)

W grafie G jest $2k$ wierzchołków i minimalny stopień wynosi k . Pokaż, że w G jest doskonałe skojarzenie (tzn. skojarzenie, w którym są wszystkie wierzchołki).

Uniwersytet Wrocławski, Informatyka

Egzamin wstępny na studia drugiego stopnia — Programowanie

18 lutego 2015

Obie kartki należy podpisać na marginesie imieniem i nazwiskiem.

Każde zadanie jest warte 20 punktów.

Czas trwania egzaminu: 60 minut.

Zadanie 1. Zaprogramuj w Prologu taki predykat `czworkami(?)`, że `cel czworkami(L)` jest spełniony wówczas, gdy `L` unifikuje się z listą, której liczba elementów jest podzielna przez cztery. Nie wolno używać żadnych predykatów standardowych ani definiować własnych predykatów pomocniczych.

Zadanie 2. Liczby jedyńkowo-dwójkowe, to ciągi cyfr 1 i 2. Ciąg $\langle c_1, \dots, c_k \rangle$, gdzie $c_i \in \{1, 2\}$, reprezentuje liczbę

$$\sum_{i=1}^k c_i \cdot 2^{i-1}.$$

Reprezentacja jest jednoznaczna. Liczbę zero reprezentuje ciąg pusty, liczbę 13 — ciąg 122 (bo z powyższej definicji wynika, że najmniej znacząca cyfra znajduje się po lewej stronie, a $1 \cdot 2^0 + 2 \cdot 2^1 + 2 \cdot 2^2 = 13$). Zaprogramuj taki predykat $\text{sum}(+, +, ?)$, że cel $\text{sum}(N, M, S)$ jest spełniony wówczas, gdy S unifikuje się listą cyfr 1 i 2 będącą reprezentacją liczby będącej sumą liczb reprezentowanych przez argumenty N i M . Predykat powinien działać deterministycznie dla podanego trybu użycia. W przypadku gdy N i M nie są listami cyfr 1 i 2 działanie predykatu może być dowolne.

Zadanie 3. Algorytm sortowania list *Bottom-Up Mergesort* polega na podzieleniu sortowanej listy na listę list jednoelementowych, a następnie na iterowaniu scalania list parami tak długo, aż pozostanie pojedyncza lista. Np. dla listy `[6,3,7,9,4,5,1,2]` kolejne etapy sortowania wyglądają następująco:

```
[ [6] , [3] , [7] , [8] , [4] , [5] , [1] , [2] , [9] ]  
  [ [3,6] , [7,8] , [4,5] , [1,2] , [9] ]  
    [ [3,6,7,8] , [1,2,4,5] , [9] ]  
      [ [1,2,3,4,5,6,7,8] , [9] ]  
        [ [1,2,3,4,5,6,7,8,9] ]
```

Zaprogramuj w Haskellu funkcje

```
merge :: Ord a => [a] → [a] → [a]
```

```
bum :: Ord a => [a] → [a]
```

Pierwsza scala dwie posortowane listy (tak jak funkcja biblioteczna o tej samej nazwie z modułu `Data.List.Utils`), druga implementuje opisany wyżej algorytm sortowania. Możesz korzystać z funkcji zdefiniowanych w preludium standardowym (`map`, `$` itp.).

Zadanie 4. Podaj typy podanych wyrażeń w Haskellu (wpisz kreskę, jeśli podane wyrażenie nie posiada typu).

<code>map map</code>	
<code>map . map</code>	
<code>flip map</code>	
<code>flip . map</code>	
<code>map flip</code>	
<code>map . flip</code>	
<code>flip flip</code>	
<code>flip . flip</code>	

gdzie

```
map _ [] = []
```

```
map f (x:xs) = f x : map f xs
```

```
(x . y) z = x (y z)
```

```
flip x y z = x z y
```