AiSD, egzamin B – 04.09.2014

1. (15p) Niech $f \prec g$ oznacza, że $f \in O(g)$ i $g \notin O(f)$ a $f \approx g$ oznacza, że $f \in O(g)$ i $g \in O(f)$. Uporządkuj następujące funkcje: $2^{(\log_2(n^2))^3}$, $2^{(\log_2(n^3))^2}$, $n\sqrt{n}$, $2^{\sqrt{n}}$.

2. (15p) Wyznacz f taką, że czas działania algorytmu znajduje się w $\Theta(f)$:
```
Hoho(int *t, int poczatek, int koniec){int suma:=0;
    for(k:=poczatek to k=koniec) suma:=suma+t[i]+ t[k];
  polowa=poczatek+(koniec-poczatek+1)/2;
  return suma + Hoho(t, poczatek, polowa)+
          Hoho(t,polowa, koniec)+ Hoho(t,poczatek+1, polowa+1; }%Hoho
```

3. (15p) Przesortuj tablicę $t[1 \ldots 8] = \{15, 12, 8, 13, 9, 8, 7, 18\}$ przy pomocy sortowania pozycyjnego (radix sort) sortując po kolejnych bitach. Przyjmij, że liczby w tablicy są zapisane binarnie i mają długość pięciu bitów.

4. (15p) Wypisz kolejne drzewa BST powstałe przez wykonanie następujących operacji na pustym drzewie: i(9), i(2), i(6), i(7), i(5), d(5), d(7). Przyjmij, że i(x) oznacza operację wstawiania do drzewa BST, która wstawia element x do korzenia drzewa. Przyjmij, że usuwając element (operacja d(x)), który nie jest liściem wpisujemy na jego miejsce odpowiedni element z jego prawego podrzewa (jeśli oba podrzewa są niepuste).

5. (15p) Napisz algorytm `MaxNieLisci(node *)`, który wywołany dla argumentu `tree` zwróci maksymalną wartość składowej key, występującej w elementach drzewa `tree`, które nie są liśćmi. Przyjmij, że `node` jest strukturą postaci `struct node{int key; node *left; node *right;}`. Pamiętaj o przypadku gdy argument funkcji jest wskazaniem pustym (null) i zwróć wtedy -1 (podobnie dla drzewa jednoelementowego).

6. (15p) Lista składa się ze struktur postaci struct{int num; char alfa;}. Trzymamy na niej elementy: (a,8), (b,4), (c,2), (d,9), (e,3). Na liście wyszukujemy albo element o danej wartości składowej alfa albo o danej wartosci składowej num. Określ, czy bardziej efektywna będzie lista uporządkowana alfabetycznie według alfa czy numerycznie według num.

   Prawdopodobieństwa wyszukiwania elementu o danej wartości składowej są podane w nawiasach: a(0.1), b(0.2), c(0.1), d(0.05), e(0.05), 2(0.2), 3(0.05), 4(0.05), 8(0.1), 9(0.1).