Bài tập

Chương 1: Các khái niệm cơ bản

1. So sánh hai hàm sau đây trong ký hiệu O:

   1a) $N\log N$ và $N^2 : N\log N = O(N^2)$ hay $N^2 = O(N\log N)$?

   1b) $N\log N$ và $N : N\log N = O(N)$ hay $N = O(N\log N)$?

2. Đưa ra đánh giá trong ký hiệu O cho các hàm sau đây

   1a) $N^2 + N\log N = O(\ldots..)$

   2b) $N + N\log N = O(\ldots..)$

   2c) $N^2 + \log N = O(\ldots..)$

3. Cho hàm

   $T(n) = 100n^{3/2} + 500n + 1000$

   xác định với đối số nguyên dương n.

   a) Chứng minh rằng $T(n) \in O(n^2)$.

   b) Chứng minh rằng $T(n) \notin \Omega(n^2)$.

4. Ta nói $f(n)$ có tốc độ tăng chậm hơn $g(n)$ nếu $f(n) = O(g(n))$. Sắp xếp các hàm sau đây theo thứ tự không giảm của tốc độ tăng (hai hàm có tốc độ tăng như nhau có thể xếp theo thứ tự tuỳ ý):

   a) $\log_2(n)$, $n^{1/2}$, $n\log_2(n)$, n^3, $\log_{10}(n)$, 10^9, $n^{3/2}$, $n^2\log_{10}(n)$, e^n, n .

   b) n^2, 2^n, $n\log n$, $n!$, $n^6 + n^2$, n^4, $n^6 - n^3$, 4^n, n, n^n, 2^{2n}.

5. Xét thuật toán đệ qui sau đây

   ALGORITHM Q(n)

   // input: số nguyên dương n

   if $n = 1$ **return** 1

   else return Q(n-1) + 2*n -1

   a) Xác định xem thuật toán tính giá trị gì: Xây dựng công thức đệ qui cho Q(n) và giải công thức đệ qui.

   b) Xây dựng công thức đệ qui đếm số phép nhân mà thuật toán phải thực hiện và giải công thức thu được.

   c) Xây dựng công thức đệ qui đếm số phép toán cộng/trừ mà thuật toán phải thực hiện và giải công thức thu được.

   d) Xây dựng công thức đệ qui cho $T(n)$ là thời gian tính của thuật toán và giải công thức.

6. Xét thuật toán sau đây để tìm khoảng cách giữa hai số gần nhau nhất trong mảng số

```
ALGORITHM MinDistance(A[0..n-1])
// Input: Mảng A[0..n-1] gồm các số nguyên
// Output: Khoảng cách nhỏ nhất giữa hai số trong mảng

dmin ← ∞
for i ← 0 to n-1 do
  for j ← 0 to n-1 do
     if i  ≠ j and |A[i] - A[j]| < dmin
        dmin  ← |A[i] - A[j]|
return dmin
```

7. Có n người. Phần tử K[i][j] của mảng K[1..n][1..n] là 1 nếu người ***i*** biết người ***j*** và là 0 nếu ngược lại. (Ta giả thiết là K[i][i] = 1 với mọi ***i***, nếu người ***i*** biết người ***j*** thì không nhất thiết người ***j*** cùng phải biết người ***i***) Một người được gọi là người đặc biệt nếu như ngoại trừ chính mình anh ta không biết ai cả nhưng tất cả những người còn lại đều biết anh ta. Ví dụ, trong hai bảng dưới đây bảng bên trái có người 3 là người đặc biệt, còn bảng bảng bên phải không có người đặc biệt.

	1	2	3	4
1	1	1	1	0
2	0	1	1	0
3	0	0	1	0
4	1	0	1	1

	1	2	3	4
1	1	1	0	1
2	0	1	1	0
3	1	0	1	0
4	1	0	1	1

Bài toán đặt ra là: Cho bảng K, hãy tìm người đặc biệt hoặc trả lời là không có người như vậy. Thuật toán trực tiếp giải bài toán đặt ra có thời gian tính $O(n^2)$. Hãy phát triển thuật toán thời gian tính $O(n)$ để giải bài toán.

2. Đánh giá thời gian tính của các hàm sau đây

a)

```
int f4(int n) {
   if (n > 1)
     return 1 + 2*f4(n-1);
   return 0;
}
```

b)

```
int f5(int n) {
   if (n > 1)
     return 3*f5(n/2) + 1
   return 0;
}
```

3. Xét thuật toán tính giá trị của $f(x,n) = x^n$ thể hiện trong hàm F(x, n) sau đây:

```
int myst(int x,int n){
   int y;
   if (n==0)return 1;
   if (n%2 == 1)
   {  y= myst(x,(n-1)/2);
      return x*y*y;
   }
   else
   {  y= myst(x,n/2);
      return y*y;
   }
}
```

a) Với giả thiết x là số nguyên, còn n là số nguyên không âm, hàm trên thực hiện công việc gì? Chứng minh khẳng định đưa ra.

b) Với giả thiết các phép toán cần thực hiện đều có thể thực hiện trong thời gian $O(1)$ và $n=2^k$, hãy đưa ra đánh giá thời gian tính của thuật toán trong ký hiệu O.

b) Với giả thiết các phép toán cần thực hiện đều có thể thực hiện trong thời gian $O(1)$ và $n=2^k$, hãy đưa ra đánh giá thời gian tính của thuật toán trong ký hiệu O.

5. Xét cấu trúc dữ liệu mô tả danh sách nối đơn:

```
struct Node{
            int Inf;
            struct Node *Next; };
typedef struct Node LIST;
```

Hãy viết chương trình con trên C

```
LIST * OddList(LIST * Linp);
```

nhận đầu vào là danh sách nối đơn với phần tử đầu tiên được trỏ bởi Linp, thực hiện việc loại bỏ tất cả các phần tử xuất hiện ở vị trí lẻ của danh sách rồi trả lại con trỏ đến đầu danh sách thu được. Giả thiết là các phần tử trong danh sách được đánh số từ 1 bắt đầu từ phần tử ở đầu danh sách.

10. Xét danh sách móc nối được mô tả bởi cấu trúc dữ liệu

```
struct Node {
    int info;
    struct Node * next;    // trỏ đến nút kế tiếp
};
```

Viết chương trình con

```
void Shuffle(Node * list);
```

nhận đầu vào là danh sách móc nối với phần tử đầu tiên được trỏ bởi **list**, thực hiện các việc sau đây:

a) Sắp xếp lại các phần tử của danh sách đã cho sao cho: các nút chẵn phải đứng trước các nút lẻ và trong trường hợp trái lại thứ tự tương đối ban đầu của các nút là không thay đổi. Một nút được gọi là nút chẵn hay lẻ nếu nó đứng ở vị trí chẵn hay lẻ trong danh sách (Vị trí của các nút trong danh sách được đánh số từ phần tử đầu tiên đến phần tử cuối cùng bắt đầu từ 1).

b) Đưa ra màn hình danh sách thu được.

Ví dụ, nếu danh sách đã cho là (11, 13, 7, 9, 3, 10) thì kết quả phải đưa ra (13, 9, 10, 11, 7, 3).

17. Đối với mỗi một trong các kiểu cấu trúc dữ liệu sau đây: Danh sách nối đơn (Singly-linked list); danh sách nối kép (Doubly-linked list); Ngăn xếp dùng mảng (Stack), Hàng đợi dùng mảng (Queue) hãy vẽ cấu trúc dữ liệu thu được sau khi lần lượt bổ sung các phần tử của dãy các khoá 4, 2, 6, **7**, 6, 5 theo thứ tự trong dãy vào các cấu trúc này (bắt đầu từ cấu trúc rỗng).

18. Hãy trình diễn cách sử dụng ngăn xếp để chuyển biểu thức dạng trung tố sau đây về dạng biểu thức hậu tố:

a) a-b*c^d+f.

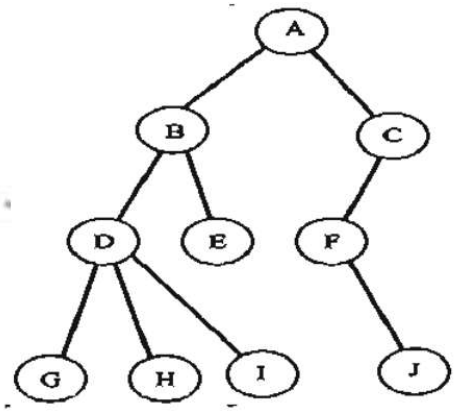
b) 1 + 2 + 3 * 4 + 5 - 6 * 7 + 8

19. Hãy trình diễn cách tính giá trị của biểu thức hậu tố sau đây nhờ sử dụng ngăn xếp:

a) 1 2 + 3 1 + * 1 1 + 1 + /

b) 3 4 + 3 5 + * **7** + 8 *

1. Xét cây cho trong hình 1.



a) Đưa ra thứ tự duyệt các nút của cây theo thứ tự trước.

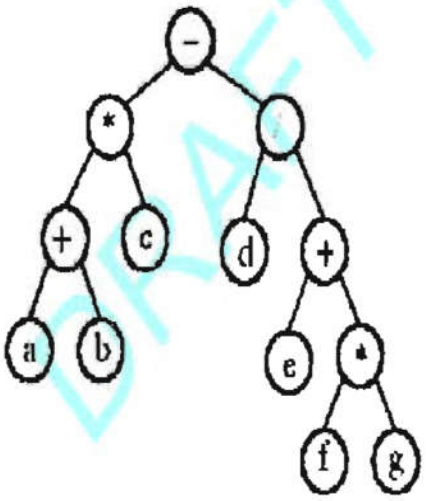
b) Đưa ra thứ tự duyệt các nút của cây theo thứ tự giữa.

c) Đưa ra thứ tự duyệt các nút của cây theo thứ tự sau.

2. Vẽ cây nhị phân độ cao 3 mà duyệt theo thứ tự sau cho ta dãy khoá là

(10, 30, 50, 20, 40, 70, 60)

6. Cho cây biểu thức trong hình dưới đây:



a) Hãy đưa ra biểu thức số học mô tả bởi cây đã cho trong ký pháp hậu tố.

b) Trình diễn thuật toán tính giá trị của biểu thức hậu tố cho trong câu 3a), trong đó a=5, b=10, c=7, d=20, e=5, f=3, g=5.

14. Xét cấu trúc dữ liệu trên C sau dây dể mô tả cây nhị phân:

```
struct Tnode
{   int key;  // Dữ liệu của nút
    struct Tnode * left;
    struct Tnode * right;
};
typedef struct Tnode treeNode;
```

Hãy viết hàm trên C

```
int countLeaft(treeNode *RootTree, int k);
```

trả lại số lượng lá của cây được trỏ bởi **RootTree**.

23. Xét cấu trúc dữ liệu mô tả cây nhị phân

```
struct TreeNode {
    int info;
    struct TreeNode * left;
    struct TreeNode * right; };
```

a) Viết hàm

```
bool IsLeft(TreeNode * root, int key);
```

nhận đầu vào là con trỏ **root** đến gốc của cây, trả lại giá trị **true** khi và chỉ khi mọi nút của cây được trỏ bởi **root** đều có trường **info** là nhỏ hơn **key**.

b) Ký hiệu n là số nút của cây. Đưa ra đánh giá thời gian tính của hàm **IsLeft** trong ký hiệu tiệm cận.

1. Trình diễn trên sơ đồ hoạt động của thuật toán sắp xếp nhanh được áp dụng để sắp xếp theo thứ tự tăng dần dãy số sau đây: **1, 2, 3, 4, 5, 6, 7, 8, 9, 10.** Giả thiết là *phần tử chốt* được chọn là *phần tử đứng cuối.*

2. Trình diễn trên sơ đồ hoạt động của thuật toán sắp xếp nhanh được áp dụng để sắp xếp theo thứ tự giảm dần dãy số sau đây: **10, 9, 8, 7, 6, 5, 4, 3, 2, 1.** Giả thiết là *phần tử chốt* được chọn là *phần tử đứng đầu.*

3. Trình diễn thuật toán sắp xếp trộn (merge sort) thực hiện sắp xếp dãy số

<22, 15, 36, 44, 10, 3, 9, 13, 29, 25>.

4. Cho dãy số

S = (23, -20, 24, -13, 28, 26, 25, 21).

Trình diễn thuật toán sắp xếp nhanh (quick sort) thực hiện sắp xếp dãy S, trong đó phần tử chốt được chọn là phần tử dứng cuối.

5. Cho mảng A = (16; 14; 10; 8; 7; 9; 3; 2; 1) biểu diễn một max-heap

a) Hãy vẽ cây nhị phân tương ứng với max-heap đã cho

b) Hãy trình bày các thao tác cần thực hiện trên cây để bổ sung thêm key=15 vào max-heap nói trên để thu được một max-heap mới.

6.

a) Vẽ cây nhị phân tương ứng với đống min (min-heap) được tạo từ mảng chứa dãy số sau đây:

40, 30, 65, 25, 35, 50, 76, 10, 28, 27, 33, 36, 34, 48, 60, 68, 80, 69.

b) Trình diễn thao tác giảm giá trị của nút với giá trị 80 xuống **13** trên min-heap thu được ở câu a)

10. Cho dãy số

46, 34, 83, 75, 25, 57, 93, 27, 17, 39, 52, 31, 93, 44

a) Vẽ *min-heap* thu được bằng cách chèn lần lượt các phần tử của dãy số vào *heap* rỗng.

b) Vẽ *max-heap* thu được bằng cách chèn lần lượt các phần tử của dãy số vào *heap* rỗng.

11. Ta có thể xây dựng đống max (max-heap) đối với dãy số A bằng cách bổ sung dần các phần tử của A vào max-heap. Xét thuật toán xây dựng theo ý tưởng vừa nêu:

```
BUILD-MAX-HEAP_Ins(A)
1     heapsize[A] = 1
2     for i = 2 to length[A] do
3       MAX-HEAP-INSERT(A, A[i])
```

Build-Max-Heap trong giáo trình:

```
Build-Max-Heap(A)
1    n = length[A]
2    for i = ⌊n/2⌋ downto 1 do
3        Max-Heapify(A, i, n)
```

```
MAX-HEAP-INSERT(A, key)
1     heapsize[A] = heapsize[A] + 1
2     i = heapsize[A]
3     A[i] = key
4       while (i > 1 and A[⌊i/2⌋] < A[i])do
5         exchange A[i] <-> A[⌊i/2⌋]
6         i = ⌊i/2⌋
```


a) Hỏi **BUILD-MAX-HEAP_Ins** và **Build_Max_Heap** trong giáo trình có luôn tạo ra cùng một đống như nhau khi làm việc với cùng một đầu vào A hay không? Nếu câu trả lời là đúng hãy chứng minh, và đưa ra phản ví dụ nếu trái lại.

b) Đưa ra đánh giá thời gian tính trong tình huống tồi nhất của **BUILD-MAX-HEAP_Ins** khi xây dựng đống gồm n phần tử.

1. a) Vẽ cây nhị phân tìm kiếm thu được bởi việc, bắt đầu từ cây rỗng, bổ sung lần lượt các khoá từ dãy khoá sau đây:

40, 30, 65, 25, 35, 50, 76, 10, 28, 27, 33, 36, 34, 48, 60, 68, 80, 69.

b) Trình diễn thao tác loại bỏ nút với khoá **30** trên cây nhị phân tìm kiếm thu được.

c) Trình diễn thao tác loại bỏ nút với khoá **65** trên cây nhị phân tìm kiếm thu được.

2. a) Vẽ cây nhị phân tìm kiếm thu được bởi việc, bắt đầu từ cây rỗng, bổ sung lần lượt các khoá từ dãy khoá sau đây:

40, 30, 65, 25, 35, 50, 76, 10, 28, 33, 36, 34, 48, 60, 68, 69, 80.

b) Trình diễn thao tác loại bỏ nút với khoá **65** trên cây nhị phân tìm kiếm thu được ở câu a).

4. a) Vẽ cây nhị phân tìm kiếm thu được khi lần lượt chèn các phần tử của dãy số nguyên

3, 1, 4, 6, 9, 2, 8, 5, 7, 0

vào cây nhị phân tìm kiếm ban đầu là rỗng. (Không cần giải thích)

b) Hãy vẽ cây nhị phân tìm kiếm thu được sau hai lần loại bỏ gốc của cây nhị phân tìm kiếm trong a). (Không cần giải thích)

5. Xét cấu trúc dữ liệu trên C để mô tả cây nhị phân tìm kiếm sau đây

```
struct TreeNode {
   float             key;
   struct TreeNode*  leftPtr;
   struct TreeNode*  rightPtr;
};
typedef struct TreeNode BSTree;
```

a) Hãy viết các hàm trên C sử dụng các cấu trúc dữ liệu trên để thực hiện các thao tác sau đây với cây nhị phân tìm kiếm:

- Tạo một nút mới:

  ```
  BSTree* makeTreeNode(float value);
  ```

- Bổ sung một nút mới vào cây nhị phân tìm kiếm:
- ```
  BSTree* insert(BSTree* nodePtr, float item);
  ```

b) Vẽ cây nhị phân tìm kiếm đối với tập các khoá S = {3, 2, 5, 4, 7, 6, 1} thu được nhờ thực hiện bổ sung lần lượt các khoá theo thứ tự đã cho vào cây nhị phân được khởi tạo ban đầu là rỗng.
  ```

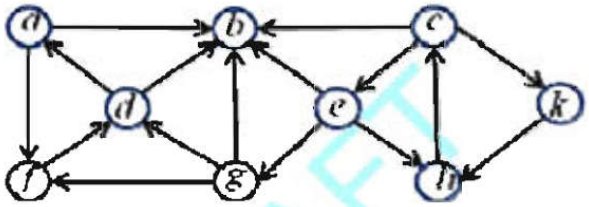
1. Xét đồ thị có hướng G với các đỉnh được đánh số từ 1 đến 7 được cho bởi ma trận kề A sau đây

$$A = \begin{vmatrix} 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{vmatrix}$$

a) Đưa ra danh sách kề của đồ thị đã cho.

b) Thực hiện tìm kiếm theo chiều sâu trên đồ thị G (giả thiết là khi duyệt các đỉnh của đồ thị và các đỉnh trong danh sách kề của một đỉnh ta duyệt theo thứ tự chỉ số tăng dần). Hãy đưa ra danh sách các cạnh của cây, cạnh ngược, cạnh tới, cạnh vòng.

3. Xét đồ thị có hướng G trong hình sau

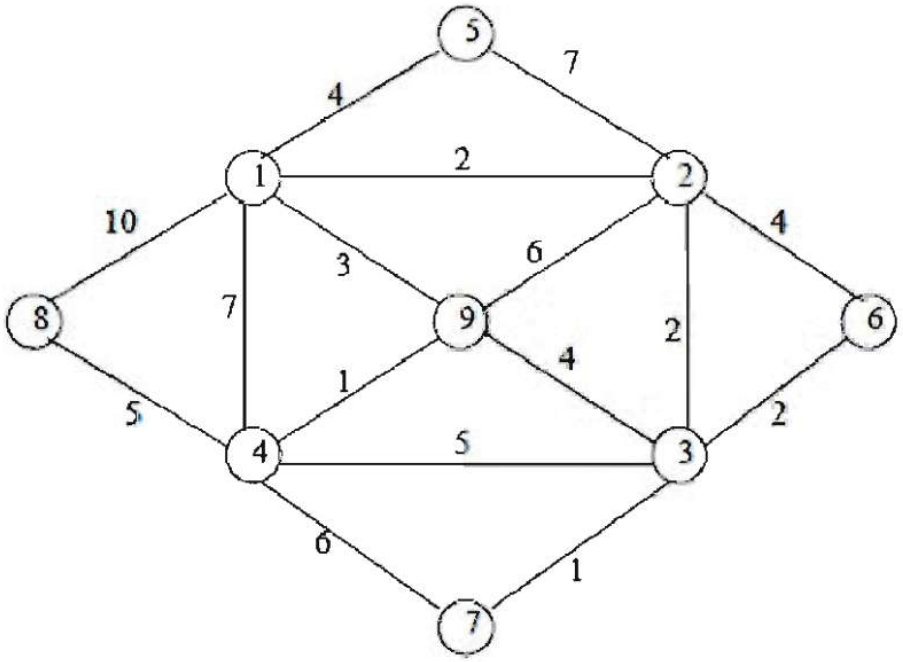


a) Đưa ra ma trận kề của đồ thị.

b) Hãy thực hiện thuật toán tìm kiếm theo chiều sâu trên G (giả thiết là khi duyệt đỉnh ta duyệt theo thứ tự trong bảng chữ cái), yêu cầu đưa ra các kết quả sau:

- Vẽ rừng DFS thu được.
- Đưa ra phân loại cạnh cho các cạnh.

6. Xét đồ thị vô hướng với trọng số trên cạnh cho trong hình vẽ sau đây:



a) Đưa ra ma trận trọng số của đồ thị.

b) Tìm cây khung nhỏ nhất của đồ thị bằng thuật toán Kruskal.