ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

ĐỀ THI MÔN: Cấu trúc dữ liệu và giải thuật
Lớp môn học: INT2203 7, 8, 9
Học kỳ I, Năm học 2011-2012

Thời gian làm bài: 120 phút
Đề thi gồm 2 trang
Sinh viên được sử dụng tài liệu in
(tuyệt đối không sử dụng máy tính và không trao đổi tài liệu)

Câu 1. Cho bảng băm địa chỉ mở giải quyết va chạm bằng phương pháp thăm dò bình phương, cỡ SIZE = 11. Từ bảng băm rỗng, sử dụng hàm băm chia lấy dư, hãy:

a) (0.5đ) đưa lần lượt các dữ liệu với khoá: 14, 0, 24, 12 , 33 , 11 , 22, 3 vào bảng băm và đưa ra bảng băm kết quả.

b) (0.5đ) loại bỏ dữ liệu với khoá là 14 rồi sau đó xen vào dữ liệu với khoá là 44 và đưa ra bảng băm kết quả.

Đáp án.

a)

0	**12**	**24**	**14**	**33**	**22**		**3**		**11**	
0	1	2	3	4	5	6	7	8	9	10

b)

0	**12**	**24**	**44**	**33**	**22**		**3**		**11**	
0	1	2	3	4	5	6	7	8	9	10

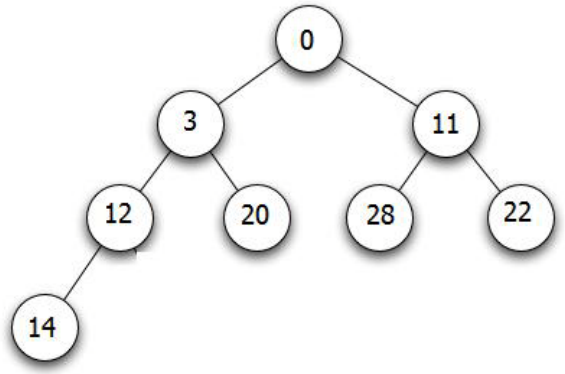
Câu 2. Cho một dãy đối tượng với các giá trị ưu tiên là 14, 0, 28 , 12 , 20 , 11 , 22, 3

a) (0.5đ) Từ cây thứ tự bộ phận rỗng, hãy xen vào từng đối tượng theo thứ tự đã liệt kê. Vẽ ra cây kết quả.

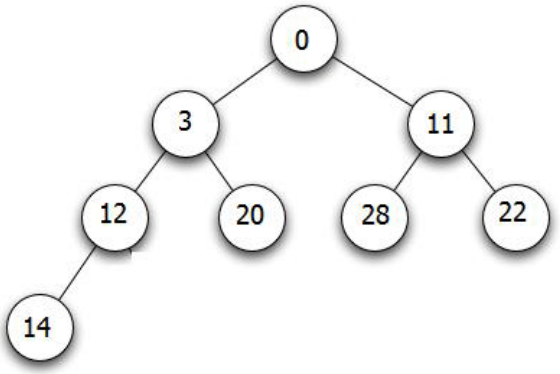
b) (0.5đ) Hãy sử dụng thuật toán xây dựng cây thứ tự bộ phận cho dãy đối tượng trên. Vẽ ra cây kết quả. So sánh cây này với cây nhận được ở câu a.

Đáp án.

a)



b)



Cây b giống cây a.

Câu 3.

a) (0.75đ) Hãy sắp xếp mảng các số {55, 22, 3, 20, 11, 90, 13, 47, 68, 36} sử dụng sắp xếp trộn (merge sort). Cần đưa ra kết quả thực hiện mỗi bước của thuật toán.

b) (0.25đ) Một thuật toán sắp xếp được xem là ổn định, nếu trật tự của các phần tử có khoá bằng nhau trong mảng đầu vào và trong mảng kết quả là như nhau. Thuật toán sắp xếp trộn có ổn định hay không?

Đáp án.

a)

55, 22, 3, 20, 11, 90, 13, 47, 68, 36
55, 22, 3, 20, 11 | 90, 13, 47, 68, 36
55, 22 | 3, 20, 11 | 90, 13, 47, 68, 36
55 | 22 | 3, 20, 11 | 90, 13, 47, 68, 36
22, 55 | 3 | 20, 11 | 90, 13, 47, 68, 36
22, 55 | 3 | 20 | 11 | 90, 13, 47, 68, 36
22, 55 | 3 | 11, 20 | 90, 13, 47, 68, 36
22, 55 | 3, 11, 20 | 90, 13, 47, 68, 36
…
3, 11, 13, 20, 22, 36, 47, 68, 55, 90

b) có (nếu cài đặt khéo)

Câu 4. Câu này hỏi về thuật toán sắp xếp nhanh (quicksort)

a) (0.5đ) Dùng giả mã hoặc mã C++, viết hàm phân hoạch (partition) mảng A[a …b] , phần tử được chọn làm chốt là phần tử đứng giữa mảng - tức là phần tử A[(a+b)/2].

b) (0.5đ) Đánh giá thời gian chạy của thuật toán sắp xếp nhanh: tốt nhất? xấu nhất? trung bình?

c) (0.5đ) Trong hàm phân hoạch, chọn chốt là phần tử chính giữa mảng có tốt hơn chọn phần tử đầu mảng hay không? Giải thích.

Đáp án.

a)

```
swap(A[a], A[(a+b)/2])
up <- a + 1
down <- b
pivot <- A[a]
while up <= down do
     while up <= down and A[up] <= pivot do up++
     while up <= down and A[down] > pivot do down–
     if up < down then swap(A[up], A[down])
swap(A[a], A[down])
```

b) tốt nhất: O(nlogn), xấu nhất: $O(n^2)$, trung bình: O(nlogn)

c) Chọn phần tử chính giữa làm chốt có tốt hơn vì ít bị ảnh hưởng bởi dữ liệu đầu vào hơn. Cụ thể là tránh được trường hợp input được sắp.

Câu 5. Theo bạn trong các cấu trúc dữ liệu liệt kê dưới đây, cấu trúc nào cài đặt hàng ưu tiên là tốt nhất? Giải thích.

a) (0.5đ) Danh sách được sắp
b) (0.5đ) Danh sách chưa sắp
c) (0.5đ) Cây tìm kiếm nhị phân
d) (0.5đ) Cây thứ tự bộ phận

Đáp án.

	insert	deleteMin	findMin	Nhận xét
Danh sách được sắp	O(n)	O(1)	O(1)	
Danh sách chưa sắp	O(1)	O(n)	O(n)	
Cây tìm kiếm nhị phân	O(h)	O(h)	O(h)	
Cây thứ tự bộ phận	O(logn)	O(logn)	O(1)	tốt nhất

Câu 6.

a) (0.5đ) Hãy cho biết dãy con chung dài nhất của 2 dãy chữ cái tạo nên xâu "HUMAN" và xâu "CHIMPANZEE". (Dãy con của một dãy là dãy thu được từ dãy gốc bằng cách xóa đi một số phần tử mà không thay đổi thứ tự giữa các phần tử còn lại.)
b) (0.5đ) Mô tả thuật toán quy hoạch động tìm dãy con chung dài nhất của 2 dãy chữ cái.
c) (0.5đ) Lập bảng lưu nghiệm của thuật toán cho ví dụ ở câu a.

Đáp án.

a) HMAN

b)

Lập bảng có mỗi hàng tương ứng với 1 chữ cái ở xâu thứ nhất (xâu a), mỗi cột tương ứng với 1 chữ cái ở xâu thứ 2 (xâu b). Bổ sung hàng đầu và cột đầu đánh dấu bắt đầu xâu. Ở ô (i, j), kí hiệu L(i,j) ta lưu độ dài của dãy con chung dài nhất giữa: dãy i chữ cái đầu của xâu a và dãy j chữ cái đầu của xâu b. Như vậy:

- L(0, 0) = 0

- với i > 0, j > 0: $L(i,j) = \begin{cases} \max\{L(i-1,j), L(i,j-1)\} \text{ nếu } a[i] \neq b[j] \\ 1 + L(i-1,j-1) \text{ nếu } a[i] = b[j] \end{cases}$

Gọi m là chiều dài xâu a, n là chiều dài xâu b. Ô (m, n) sẽ lưu độ dài của dãy con chung dài nhất của 2 dãy chữ cái.

Từ đây ta có thể xây dựng dãy con chung dài nhất của a và b như sau:
Xuất phát từ ô i = m, j = n:
Nếu a[i] = b[j], push a[i] vào stack, i--, j--
Nếu a[i] ≠ b[j]
    nếu L(i,j) = L(i-1,j) thì i--
    nếu L(i,j) = L(i,j-1) thì j--
Lặp tới khi i = 0, j = 0.
Dãy con dài nhất thu được khi ta pop toàn bộ stack ra.
c)

		C	H	I	M	P	A	N	Z	E	E
	0	0	0	0	0	0	0	0	0	0	0
H	0	0	1	1	1	1	1	1	1	1	1
U	0	0	1	1	1	1	1	1	1	1	1
M	0	0	1	1	2	2	2	2	2	2	2
A	0	0	1	1	2	2	3	3	3	3	3
N	0	0	1	1	2	2	3	4	4	4	4

Câu 7. Hãy viết đoạn chương trình C++ giải quyết các vấn đề sau

a) (0.5đ) In ra màn hình số nguyên ngẫu nhiên trong khoảng từ 0 đến N. Chú ý là đoạn chương trình cần sinh ra các số ngẫu nhiên khác nhau ở các lần chạy khác nhau.

b) (1.5đ) Hàm thành viên enqueue của lớp ArrayQueue chèn thêm giá trị x vào hàng đợi số nguyên cài bằng mảng cấp phát động. Chú ý: f là chỉ số của ô đầu hàng đợi, r là chỉ số của ô liền sau ô cuối hàng đợi (nghĩa là ô liền sau ô cuối hàng đợi sẽ không lưu dữ liệu). Bạn được cho khai báo lớp ArrayQueue như bên dưới.

```
#define INIT_CAPACITY 2
class ArrayQueue {
public:
      // hàm kiến tạo
      ArrayQueue(){
            capacity = INIT_CAPACITY;
            element = new int[capacity];
            f = 0;
            r = 0;
      }
```

```
	// hàm hủy
	~ArrayQueue(){
		delete [] element;
		element = NULL;
		capacity = 0;
		f = 0;
		r = 0;
	}

	void enqueue(int x);// ham them x vao cuoi hang doi
private:
	int * element;// con tro toi dau mang
	int capacity;// kich thuoc toi da cua mang
	int f;// chi so cua o dau hang doi
	int r;// chi so cua o lien sau o cuoi hang doi
};
```

Đáp án.

a)

```
srand(time(0));
cout << rand() % (N + 1);
```

b)

Bổ sung hàm isFull kiểm tra hàng đợi đầy

```
bool ArrayQueue::isFull(){
	int size = (capacity - f + r) % capacity;
	return size == capacity - 1;
}
```

```
// ham them x vao cuoi hang doi
void ArrayQueue::enqueue(int x){
	// neu day thi phai noi mang + sao du lieu tu sang mang moi
	if(isFull()){
		int * temp = new int[capacity * 2 + 1];
		// sao.......
		for(int i = f, j = 0; j < capacity - 1; i = (i + 1) % capacity, j++)
			temp[j] = element[i];
		delete[] element;
		f = 0;
		r = capacity - 1;
		element = temp;
		capacity = capacity * 2 + 1;
	}

	element[r] = x;
	r = (r + 1) % capacity;
}
```