MA/CSSE 473 Day 23 Announcements and Summary

Announcements:

1. HW 10 is due today.
2. HW 11 is due next Thursday (it is a very long assignment, and I may add another problem or two, so you have a week to do it).
3. B-trees (section 7.4 in Levitin should be straightforward for those who have had experience with other balanced trees, so I am asking you to read this section on your own and ask questions about anything you do not understand.
4. Exam 2 Tuesday Nov 4 in class
5. In my office today: hours 6, 7, 8.

Main ideas from today:

1. List several examples where using additional space allows us to solve a problem faster.

Background for next three parts: http://www.rose-hulman.edu/class/csse/csse230/201230/Slides/17-Graphs-HashTables.pdf

2. (with partners) Quickly discuss these issues.
What problem do we try to solve by hashing?
What is the general idea of how hashing works?
Why does it fit into Chapter 7 (space-time tradeoffs)?
What are the main issues to be addressed when discussing hashing implementation?
How to choose between a hash table and a binary search tree?

3. Discuss with partners to be sure that you can define these terms:
hash function, collision, load factor (λ) , perfect hash function, open addressing, linear probing, cluster, quadratic probing, rehashing, separate chaining. When doing an insertion into the table, how does the value of λ affect expected number of probes If you need hints, see the PowerPoint slides at
http://www.rose-hulman.edu/class/csse/csse230/201230/Slides/17-Graphs-HashTables.pdf

4. (with partners again) Suppose we have a table size of 19, the identity function h(key)=key as the hash function, and use linear probing for collision resolution. The following numbers are inserted in the table in the given order. Beside each number in the left table, write the array index where the value ends up, and the number of times we will have to probe the table before we find the correct place to insert that number (the number of probes will always be at least 1). You can use the second set of tables to help you do this.
For the middle table, do the same thing, but with quadratic probing.
For the right table, use double hashing (rehashing) adding h2(key) = key % 4 for each new probe.

linear			quadratic			rehash			
Value	index	probes	value	index	probes	value	index	probes	
15	15	1	15	15	1	15	15	1	
34	16	2	34	16	2	34	17	2	h2=2
19			19			19			
40			40			40			
1			1			1			
14			14			14			
18			18			18			
39			39			39			
35			35			35			
71			71			71			

index	key	index	key	index	key
0		0		0	
1		1		1	
2		2		2	
3		3		3	
4		4		4	
5		5		5	
6		6		6	
7		7		7	
8		8		8	
9		9		9	
10		10		10	
11		11		11	
12		12		12	
13		13		13	
14		14		14	
15		15		15	
16		16		16	
17		17		17	
18		18		18	

4. Show by contradiction that if p is a prime number larger than 3, if i and j are $\le \lfloor p/2 \rfloor$, and if $i \neq j$, then $H + i^2 \not\equiv H + j^2 \pmod{p}$.

5. In string search algorithms, to what do the terms *pattern* and *text* refer?

   pattern:

   text:

6. What is the main source of inefficiency in brute force string searching?

7. What is unusual about the comparison order in Horspool's string search algorithm?

8. If the text character that we compare to the last pattern character occurs nowhere in the pattern, how far right can we shift the pattern, and still be sure that we do not miss a match?