CSE 373

MAY 1ST – HASHING CONCLUSION

EXAM RESULTS

- **Overall, you did very well**
  - Average in the mid-80s
- **ADT vs Data Structure**
- **Algorithm Analysis**
- **Rigor for final exam**

EXAM RESULTS

- **If you did poorly on this exam,**
  - That's okay, but we should talk about what we can do to help with your performance
- **Midterm scores were high so we'll speed up a little bit, and expect the final exam to be more difficult**

GRADE MINUTIAE

- **I will have caught up on regrade requests by Wednesday, at that point, all grades from the first half of the quarter will be finalized.**
- **Exam regrades**
  - You can discuss compaints with the TAs, but all points will go through me.
  - Extra office hours on Friday (1:00-2:30) and (3:30-5:00) for this purpose

GRADE MINUTIAE

- **End of Quarter HW make up**
  - At the end of week 7, I will release two extra credit assignments
  - One will be a complex data structure that you will need to implement and test
  - The other will be a write-up about some algorithm or method
  - You may complete one of these two to replace your lowest grade on HW for either code or a writeup.

HOMEWORK 4

- Homework 4 will be on graphs and will come out after we introduce graphs to you on Wednesday
- HW 5 will use the implementation of your graph from this HW, so please make sure your implementation is working well
- You will receive feedback from HW4 before HW5 is due to make sure everything is working

TODAYS LECTURE

- Quadratic Probing
- Separate Chaining

HASHING

- **Review**
  - Hashtables have two important parts
    - Hash function
    - Array storage

HASHING

- **Hash function**
  - Maps the large subject domain onto the small set of relevant data.
  - For example, H(x) = x % tablesize
  - The function should run in constant time
  - It should distribute data evenly throughout the table

HASHING

- **Array storage**
  - Array of data that the hash function maps onto
  - The more full the array is, the higher the chances for a collision
  - Direct relation between memory efficiency and runtime efficiency

HASHING

- **Collisions**
  - A collision is when two keys map to the same index in the array.
  - Right now, our strategy is linear probing,
  - Go through the data structure in linear order until a hole is found
  - Guaranteed to place data if there is room, but runtime can be bad if there is significant *clustering*

HASHING

- **Clustering**
  - When a large chunk of the table becomes a single block of data, all searches and inserts must iterate through the entire cluster to find an opening
  - What are some possible improvements to linear probing?

HASHING

- **Quadratic probing**
  - Whereas linear probing increments the index by one each time, quadratic probing goes through the squares
  - For example, linear probing would check index 3, then 3+1, then 3+2, then 3+3, then 3+4 and so forth
  - Quadratic probing would check index 3, then index 3+1, then 3+4 then 3+9 then 3+16

HASHING

- **Quadratic probing**
  - An advantage is that it does not form linear clusters (**primary clustering**), however there are other downsides

QUADRATIC PROBING

- Let our hash function for ints, H(x) = x%7
- Insert, 3, 10,17,24,31,38

0	
1	
2	
3	
4	
5	
6	

QUADRATIC PROBING

- Let our hash function for ints, H(x) = x%7
- Insert, 3, 10,17,24,31,38
- What happens?

0	
1	
2	
3	
4	
5	
6	

QUADRATIC PROBING

- Let our hash function for ints, H(x) = x%7
- Insert, 3, 10,17,24,31,38
- What happens?

0
1
2
3: 3
4
5
6

QUADRATIC PROBING

- Let our hash function for ints, H(x) = x%7
- Insert, 3, 10,17,24,31,38
- What happens?

0
1
2
3: 3
4:10
5
6

QUADRATIC PROBING

- Let our hash function for ints, H(x) = x%7
- Insert, 3, 10,17,24,31,38
- What happens?

0: 17
1
2
3: 3
4:10
5
6

QUADRATIC PROBING

- Let our hash function for ints, H(x) = x%7
- Insert, 3, 10,17,24,31,38
- What happens?

0: 17
1
2
3: 3
4:10
5: 24
6

QUADRATIC PROBING

- Let our hash function for ints, H(x) = x%7
- Insert, 3, 10,17,24,31,38
- What happens?
- Where does 31 go?
  - 31%7 = 3

0: 17
1
2
3: 3
4:10
5: 24
6

QUADRATIC PROBING

- **Let our hash function for ints, H(x) = x%7**
- **Insert, 3, 10,17,24,31,38**
- **What happens?**
- **Where does 31 go?**
  - 31%7 = 3
  - 3+1%7 = 4

0: 17
1
2
3: 3
4:10
5: 24
6

QUADRATIC PROBING

- Let our hash function for ints, H(x) = x%7
- Insert, 3, 10,17,24,31,38
- What happens?
- Where does 31 go?
  - 31%7 = 3
  - 3+1%7 = 4
  - 3+2%7 = 5

0: 17
1
2
3: 3
4:10
5: 24
6

QUADRATIC PROBING

- Let our hash function for ints, H(x) = x%7
- Insert, 3, 10,17,24,31,38
- What happens?
- Where does 31 go?
  - 31%7 = 3
  - 3+1%7 = 4
  - 3+2%7 = 5
  - 3+4%7 = 0

0: 17
1
2
3: 3
4:10
5: 24
6

QUADRATIC PROBING

- Let our hash function for ints, H(x) = x%7
- Insert, 3, 10,17,24,31,38
- What happens?
- Where does 31 go?
  - 31%7 = 3
  - 3+1%7 = 4
  - 3+2%7 = 5
  - 3+4%7 = 0
  - 3+9%7 = 5

0: 17
1
2
3: 3
4:10
5: 24
6

QUADRATIC PROBING

- Let our hash function for ints, H(x) = x%7
- Insert, 3, 10,17,24,31,38
- What happens?
- Where does 31 go?
  - 31%7 = 3
  - 3+1%7 = 4
  - 3+2%7 = 5
  - 3+4%7 = 0
  - 3+9%7 = 5
  - 3+16%7 = 5

0: 17
1
2
3: 3
4:10
5: 24
6

QUADRATIC PROBING

- **Let our hash function for ints, H(x) = x%7**
- **Insert, 3, 10,17,24,31,38**
- **What happens?**
- **Where does 31 go?**
  - 31%7 = 3
  - 3+1%7 = 4
  - 3+2%7 = 5
  - 3+4%7 = 0
  - 3+9%7 = 5
  - 3+16%7 = 5
  - 3+25%7 = 0
  - 3+36%7 = 4
  - 3+49%7 = 0
  - 3+64%7 = 4

0: 17
1
2
3: 3
4:10
5: 24
6

QUADRATIC PROBING

- **This is called secondary clustering**
  - Even when there is space available in the table, quadratic probing is not guaranteed to find an opening
  - In fact, half the array has to be empty to guarantee an opening
  - This approach reduces the O(n) problem of linear probing, but it introduces even larger memory constraints

SECONDARY HASHING

- **The final probing method uses a secondary hash function**
  - If H(x) and H(y) both point to the same index, then we increment by some secondary hash value F(y) each time we need to find a new position
  - Obviously, F(y) cannot be a multiple of the table size, or else the location will never move

CHAINING

- These are the probing techniques
- However, if we allow two keys to occupy the same spot in the table, this is called chaining
- Chaining will always find a place for data, but it can get to O(n) runtime if the table isn't resized
- Resizes are costly!

NEXT CLASS

- Finish our discussion of hash tables and chaining
- Introduce a new abstract structure called the graph
- Most of the rest of the course will be on graphs