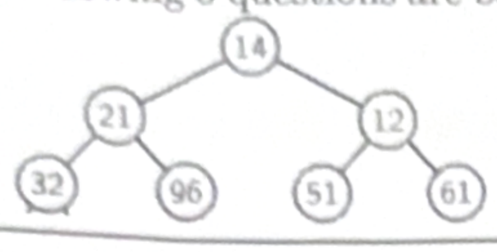
Dylan Reed

COMP-2540: MIDTERM EXAM 2 (2025)

The duration of the exam is 75 Minutes. This is a close-book exam. You are not allowed to use notes, calculators, computers, or cell phones. Please select the **most appropriate** answer for each question. Remember to fill in the scantron sheet using #2 **pencil**. There are 74 questions, 7 pages in total.

Tree traversal

Following 5 questions are based on the tree below



32 96 21 51 61 12 14
14 21 12 32 96 51 61

Q1. Which of the following is NOT true?

(a) It is a heap
(b) It is full tree
(c) It is a proper tree
(d) It is a complete tree

Q2. Which of the following is in-order traversal?

14 21 32 96 12 51 61

(a) 32,21,96,14,51,12,61
(b) 14,21,32,96,12,51,61
(c) 32,96,21,51,61,12,14
(d) 14,21,12,32,96,51,61
(e) 61,51,96,32,12,21,14

Q3. Which is pre-order traversal?

(a) 32,21,96,14,51,12,61
(b) 14,21,32,96,12,51,61
(c) 32,96,21,51,61,12,14
(d) 14,21,12,32,96,51,61
(e) 61,51,96,32,12,21,14

Q4. Which is post-order traversal?

(a) 32,21,96,14,51,12,61
(b) 14,21,32,96,12,51,61
(c) 32,96,21,51,61,12,14
(d) 14,21,12,32,96,51,61
(e) 61,51,96,32,12,21,14

Q5. Which of the following is the breadth first traversal

(a) 32,21,96,14,51,12,61
(b) 14,21,32,96,12,51,61
(c) 32,96,21,51,61,12,14
(d) 14,21,12,32,96,51,61
(e) 61,51,96,32,12,21,14

Linear probing

The following 4 questions are based on the following Hash table of size N = 9 that is represented by an array a. Suppose that the hash function $h(n) = n \ mod \ 9$, and we use linear probing to solve collision.

a=			11		31			61	80
	0	1	2	3	4	5	6	7	8

Q6. If we insert key 6, it will be inserted at

(a) a[5]
(b) a[6]
(c) a[7]
(d) a[0]
(e) a[1]

Q7. After that we insert key 15, it will be inserted at

(a) a[5]
(b) a[6]
(c) a[7]
(d) a[0]
(e) a[1]

Q8. After that we insert key 16, it will be inserted at

(a) a[5]
(b) a[6]
(c) a[7]
(d) a[0]
(e) a[1]

Q9. Normally, when resizing should happen in the above insertion process?

(a) When insert 6
(b) when insert 15
(c) when insert 16
(d) No resizing

Q10. In the implementation of hash map, the load factor is defined as n/N, where N is the size of the hash table, and n is the number of elements. If the collision is handled by open addressing, the load factor should be

(A) greater than the load factor in separate chaining
(B) smaller than the load factor in separate chaining
(C) equal to the load factor in separate chaining
(D) equal to 1.

Q11. In HashMap, the average case time complexity for the put(key, value) operation is

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$

Q12. What is the worst case time complexity of the put(k, v) operation in HashMap?

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$

Q13. In the context of hashing, Horner's Method is used to compute:

(a) The load factor of a hash table
(b) The hash value of a string efficiently
(c) The number of collisions in a hash table
(d) The maximum size of a hash table

Heap Construction

The following 6 questions of based on the following array for the heap representation problem discussed in class. Suppose that we want to construct a max-Heap.

0	1	2	3	4	5	6	7	8
26	21	12	32	96	51	61	28	

In HashMap, the average case time complexity for the put(key, value) operation is

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$

2. What is the worst case time complexity of the `put(k,v)` operation in HashMap?

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$

13. In the context of hashing, Horner's Method is used to compute:

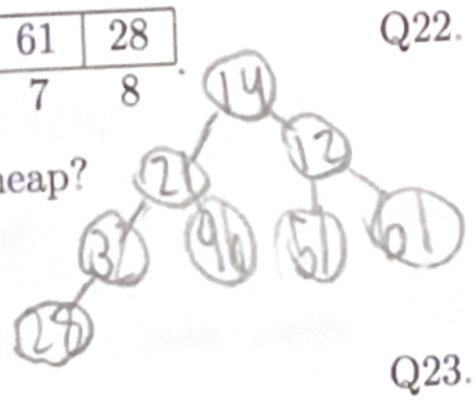
(a) The load factor of a hash table
(b) The hash value of a string efficiently
(c) The number of collisions in a hash table
(d) The maximum size of a hash table

Heap Construction

The following 6 questions of based on the following array for the heap representation problem discussed in class. Suppose that we want to construct a max-Heap.

a=

	14	21	12	32	96	51	61	28
0	1	2	3	4	5	6	7	8



Q14. How many elements are there in the heap?

(a) 7
(b) 8
(c) 9
(d) 10

Q15. What is the height of the corresponding tree?

(a) 2
(b) 3
(c) 4
(d) 5

Q16. What is the time complexity to heapify the array?

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$

Q17. During heapification, which is the first element `sink` method will run on?

(a) 14
(b) 12
(c) 32
(d) 96
(e) 28

Q18. During heapification, which is the first element will be swapped?

(a) 14
(b) 21
(c) 12
(d) 32
(e) 28

Q19. Which series is used to derive the time complexity?

(a) $1+2+3+\cdots+n$
(b) $\frac{1}{2}+\frac{2}{2^2}+\frac{3}{2^3}+\cdots+\frac{1}{2^n}$
(c) $\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}$
(d) $2^0+2^1+\cdots+2^n$

Q20. The time complexity of HeapSort on an array of n elements is

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$

Q21. Comparing merge sort and heap sort, which of the following is true?

(a) Heap sort has better time complexity
(b) Merge sort has better time complexity
(c) Heap sort uses less space
(d) Merge sort uses less space

Q22. Given a heap with 64 nodes, the height of the heap is

(a) 3
(b) 4
(c) 5
(d) 6
(e) 64

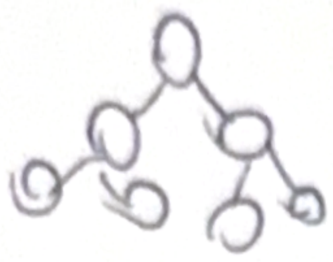
Q23. By reading the following `swim` method, we know that it implements

```
void swim(int k) {
   while (k > 1 && less(k, k/2)) {
      exch(k, k/2);
      k = k/2;
   }
}
```

(a) a minHeap
(b) a maxHeap
(c) a Hash table
(d) a priority queue

Q24. Given a function $f(n) = n^2 + 3n$. Which of the following is not true for $f(n)$?

(a) it is $O(n^2)$
(b) it is $O(n^3)$
(c) it is $O(n)$
(d) it is $\Theta(n^2)$



. What is the time complexity of the following algorithm, supposing that the tree has n nodes:

```
int height(Node node) {
  int h = 0;
  for (Node c: children(node))
      h = Math.max(h, 1+height(c));
  return h;
}
```

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$

Q26. Given a proper binary tree. Let N_e denote the number of external nodes, and N_i the number of internal nodes. Then

(a) $N_i = N_e + 1$
(b) $N_i = N_e - 1$
(c) $N_i = 2N_e$
(d) $2N_i = N_e$

Q27. In amortized analysis of the Stack push operation using dynamic array. Which series is used for inferring the time complexity when resizing using constant k increment?

(a) $1+2+3+\cdots+n$
(b) $\frac{1}{1}+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}$
(c) $\frac{1}{2^0}+\frac{1}{2^1}+\frac{1}{2^2}+\cdots+\frac{1}{2^n}$
(d) $2^0+2^1+\cdots+2^n$

Q28. In Stack ADT, the operation that has the highest time complexity is

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$

Q29. In the Queue ADT discussed in class, the most costly operation has time complexity

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$

Q30. In the List ADT discussed in class, the most costly operation has time complexity

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$

Q31. In a proper binary tree, a node can have how many children?

(a) 0 or 1
(b) 0 or 2
(c) 1 or 2
(d) always 2

Q32. In the dynamic array implementation of Stack, if we choose to increase the array length by a constant every time more space is needed, the time complexity of n push operation is

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$

Q33. Given the following code for an application of Stack.

```
boolean x(String expression) {
  final String opening  = "({[";
  final String closing  = ")}]";
  Stack<Character> buffer = new Stack<>();
  for (char c : expression.toCharArray()) {
    if (opening.indexOf(c) != -1)
        buffer.push(c);
    else if (closing.indexOf(c) != -1) {
      if (buffer.isEmpty()) return false;
      if (closing.indexOf(c)
        !=opening.indexOf(buffer.pop()))
        return false;
    }
  }
  return buffer.isEmpty();
}
```

(a) It reverses the input string
(b) it checks whether the expression is a valid arithmetic expression
(c) it returns `true` for input `(x+y)()`
(d) it returns `false` for input `(x+y)()`

Q34. A binary tree with n nodes will have

(A) n edges
(B) n-1 edges
(C) n/2 edges
(D) n+1 edges
(E) None of the above

Q35. Given a tree with n nodes. Its height is always smaller than

(a) n/2
(b) log n
(c) n log n
(d) n

Q36. Given a binary tree with n nodes. The height is always greater than

(A) n/2
(B) log n -1

(C) n log n
(D) n-1

$\log(x^2y)$ is equal to

(a) $2\log(x)\log(y)$
(b) $\log(2x)+\log(y)$
(c) $2+\log(x)+\log(y)$
(d) $2\log(x)+\log(y)$

. Among the program we discussed in class, which of the following is a tail recursive program?

(A) Factorial
(B) Linear sum
(C) Binary sum
(D) Reverse array

39. Given the following recursive program,

```
boolean x(int[] data,int target,int low,int high) {
  if (low>high) return false;
  int mid = (low + high) / 2;
  if (target == data[mid]) return true;
  if (target < data[mid])
      return x(data, target, low, mid - 1);
  return x(data, target, mid + 1, high);
}
```

It's recurrence relation is

(A) $T(n)=T(n-1)+1$
(B) $T(n)=T(n-1)+n$
(C) $T(n)=T(n/2)+1$
(D) $T(n)=2T(n/2)+n$
(E) $T(n)=T(n-1)+T(n-2)+1$

Q40. For the above program, it's time complexity is

(A) $O(n)$
(B) $O(n^2)$
(C) $O(\log n)$
(D) $O(n\log n)$
(E) $O(2^n)$

Q41. The following sequence is equal to

$$1+2+3+\cdots+n=$$

(a) 2n
(b) n^2
(c) $n^2/2$
(d) $n^2/2+n/2$
(e) $n(n-1)/2$

Q42. Which is the closest to the following sequence

$$\frac{1}{1}+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}$$

(a) $2n$
(b) $n\log n$
(c) $\log n$
(d) $\ln n$
(e) n

Q43. The following sequence is equal to

$$1+2^1+2^2+\cdots+2^{n-1}$$

(a) 2^n
(b) n^2
(c) n^2-1
(d) n^2+n
(e) 2^n-1

Q44. Given the following algorithm to find the max element from an array.

```
int arrayMax(int[] data) {
  int n=data.length;
  int currentMax=data[0];
  for (int j=1;j<n;j++)
       if(data[j]>currentMax)
           currentMax=data[j];
  return currentMax;
}
```

How many times the assignment statement

$currentMax = data[j]$ is executed on average?

(a) n
(b) $n/2$
(c) $\ln n$
(d) $n\log n$
(e) $\log n$

Q45. Which of the following is a disadvantage of using arrays over linked lists?

(A) Arrays require contiguous memory allocation.
(B) The size of an array is fixed after declaration.
(C) Insertion and deletion at arbitrary positions are slow.
(D) All of the above.

Q46. What is the time complexity of searching for an element in an unsorted array of size n?

(A) $O(1)$
(B) $O(\log n)$
(C) $O(n)$
(D) $O(n\log n)$
(E) $O(n^2)$

Q47. What is the time complexity of searching for an element in a sorted array of size n using binary search?

(A) $O(1)$
(B) $O(\log n)$
(C) $O(n)$
(D) $n\log n$
(E) $O(n^2)$

What is the worst-case time complexity of deleting the tail node in a singly linked list with n nodes?

(A) $O(1)$
(B) $O(\log n)$
(C) $O(n)$
(D) $O(n \log n)$
(E) $O(n^2)$

9. What is the time complexity of the following program:

```
public static boolean x(int[ ] data) {
  int n = data.length;
  int[ ] temp = Arrays.copyOf(data, n);
  Arrays.sort(temp);
  for (int j=0; j<n-1; j++)
    if (temp[j] == temp[j+1])
      return false;
  return true;
}
```

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$

Q50. what is the function of the code above?

(a) Search for an element
(b) Check whether there are duplicate data items
(c) Sort an array
(d) Check whether an array is sorted

Q51. What is the time complexity for the following code:

```
boolean x(int [] groupA, int [] groupB, int [] groupC){
  for (int a : groupA)
    for (int b : groupB)
      if (a == b)
        for (int c : groupC)
          if (a == c)
            return false;
  return true;
}
```

(A) $O(\log n)$
(B) $O(n)$
(C) $O(n \log n)$
(D) $O(n^2)$
(E) $O(n^3)$

Q52. What is the time complexity for the following program:

```
public static String aFunction(char c, int n) {
  String answer = "";
  for (int j=0; j < n; j++)
    answer += c;
  return answer;
}
```

(A) $O(1)$
(B) $O(\log n)$
(C) $O(n)$
(D) $O(n \log n)$
(E) $O(n^2)$

Q53. Given the following merge sort code, which line has an error?

```
1  <K>void mergeSort(K[] S,Comparator<K> comp){
2    int n = S.length;
3    if (n < 2) return;
4    int mid = n/2;
5    K[] S1 = Arrays.copyOfRange(S, 0, mid);
6    K[] S2 = Arrays.copyOfRange(S, mid+1, n);
7    mergeSort(S1, comp);
8    mergeSort(S2, comp);
9    merge(S1, S2, S, comp);
10 }
```

(A) 3
(B) 4
(C) 5
(D) 6
(E) 7

Q54. Which line of the following code will cause infinite loop?

```
1  int x(int n) {
2    if (n==0)
3      return 1;
4    else
5      return n * x(n);
6  }
```

(A) 1
(B) 2
(C) 3
(D) 4
(E) 5

Q55. Which of the following operations on an array has a time complexity of $O(1)$ in the worst case?

(A) Searching for an element.
(B) Deleting an element.
(C) Inserting an element at a given index.
(D) Accessing an element at a specific index.

Q56. Which of the following is true for doubly linked lists but not for singly linked lists?

(A) They allow traversal in both forward and backward directions.
(B) Each node has a pointer to the next node.
(C) Insertion at the head is $O(1)$.
(D) Deletion of the head node takes $O(1)$ time.

What is the time complexity of inserting an element into an unsorted array of size n?

(A) $O(1)$
(B) $O(\log n)$
(C) $O(n)$
(D) $n \log n$
(E) $O(n^2)$

In a singly linked list discussed in class:

(A) Insertion at the tail must be done in $O(n)$
(B) Removing at the tail can be done in $O(1)$
(C) Removing at the head can be done in $O(1)$
(D) Finding the node in the middle of the list takes $O(\log n)$

. Given the recurrence relation $T(n) = 2T(n/2) + n$. What is the closed-form upper bound solution for $T(n)$?

(A) $O(n)$
(B) $O(n^2)$
(C) $O(n \log n)$
(D) $O(\log n)$
(E) $O(2^n)$

60. Given the recurrence relation $T(n) = T(n/2) + 1$. What is the closed-form upper bound solution for $T(n)$?

(A) $O(n)$
(B) $O(n^2)$
(C) $O(n \log n)$
(D) $O(\log n)$
(E) $O(2^n)$

Q61. Given the recurrence relation $T(n) = 2T(n-1)+1$. What is the closed-form upper bound solution for $T(n)$?

(A) $O(n)$
(B) $O(n^2)$
(C) $O(n \log n)$
(D) $O(\log n)$
(E) $O(2^n)$

Q62. Given the recurrence relation $T(n) = T(n-1)+1$. What is the closed-form upper bound solution for $T(n)$?

(A) $O(n)$
(B) $O(n^2)$
(C) $O(n \log n)$
(D) $O(\log n)$
(E) $O(2^n)$

The next 3 questions are based on the following code.

```
long x(int n) {
    if (n <= 1)      return n;
    return x(n-2) + x(n-1);
}
```

Q63. What is the recurrence relation for the running time of the program?

(a) $T(n) = T(n-1) + 1$
(b) $T(n) = T(n-1) + n$
(c) $T(n) = 2T(n/2) + 1$
(d) $T(n) = 2T(n/2) + n$
(e) $T(n) = T(n-1) + T(n-2) + 1$

Q64. The time complexity is

(a) Linear
(b) Quadratic
(c) Exponential
(d) Logarithmic

Q65. The best algorithm for the same problem can have what kind of time complexity?

(a) Linear
(b) Quadratic
(c) Exponential
(d) Logarithmic

The following 3 questions are based on the following code:

```
int x(int[] data, int low, int high) {
  if (low > high)    return 0;
  if (low==high)     return data[low];
  int mid=(low + high) / 2;
  return x(data,low,mid)+x(data,mid+1,high);
}
```

Q66. The function of the code is to

(a) sort an array
(b) search for an element in an array
(c) find the minimal element
(d) summation of the array

Q67. It's time complexity is

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$

It's recurrence relation for the running time is

(a) $T(n) = T(n-1) + 1$
(b) $T(n) = T(n-1) + n$
(c) $T(n) = 2T(n/2) + 1$
(d) $T(n) = 2T(n/2) + n$
(e) $T(n) = T(n-1) + T(n-2) + 1$

. What is the time complexity of the Selection Sort algorithm, supposing that the data size is n?

(A) $O(n)$
(B) $O(n^2)$
(C) $O(\log n)$
(D) $O(n \log n)$
(E) $O(2^n)$

70. What is the time complexity of the Insertion Sort algorithm, supposing that the data size is n?

(A) $O(n)$
(B) $O(n^2)$
(C) $O(\log n)$
(D) $O(n \log n)$
(E) $O(2^n)$

Q71. What is the time complexity of the following recursive function?

```
int someFunction(int n) {
    if (n == 0) return 1;
    else  return n * someFunction(n-1);
  }
```

(A) $O(n)$
(B) $O(2^n)$
(C) $O(n^2)$
(D) $O(\log n)$
(E) $O(n \log n)$

Q72. What is the time complexity of the following function?

```
public static long[] fibonacci(int n) {
  if (n <= 1) {
    long[] answer = {n, 0};
    return answer;
  } else {
    long[] temp = fibonacci(n - 1);
    long[] answer = {temp[0] + temp[1], temp[0]};
    return answer;
  }
}
```

(A) $O(1)$
(B) $O(n \log n)$
(C) $O(n)$
(D) $O(n^2)$
(E) $O(2^n)$

Q73. What is the time complexity of the following recursive function?

```
int x(int n):
    if (n <= 1) return 1;
    return 3 * x(n - 1)
```

(A) $O(n \log n)$
(B) $O(3^n)$
(C) $O(n)$
(D) $O(n^3)$
(E) $O(\log n)$

Q74. Given the following function for calculating the height of a tree. Suppose the tree has n number of nodes. The time complexity of the function is

```
int height (Node node){
  int h = 0;
  for ( Node node: nodes())
    if (isExternal(node))
      h = Math.max(h, depth(node)) ;
  return h ;
}
```

(a) $O(1)$
(b) $O(\log n)$
(c) $O(n)$
(d) $O(n \log n)$
(e) $O(n^2)$