This problem set has 16 questions, for a total of 110 points. Answer the questions below and mark your answers in the spaces provided. If the question asks for showing your work, you must provide details on how your answer was calculated.

Your Name: _____

1. [5 points] Which of the following descriptions best describes what **mystery** does?

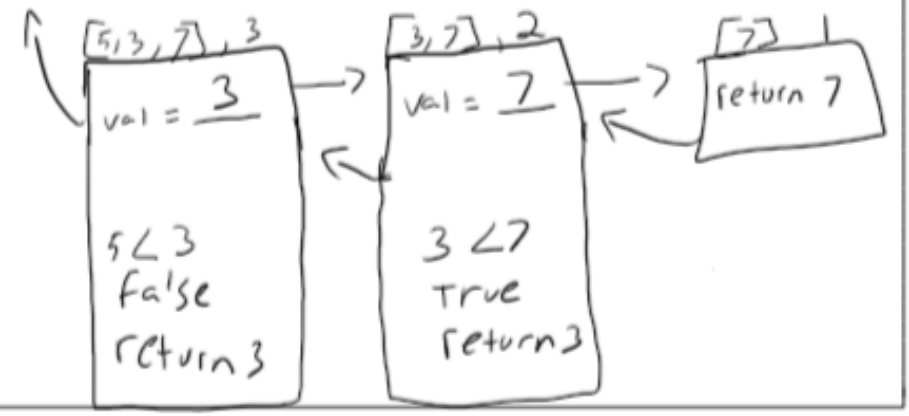
```
int mystery(int *arr, int n) {
    if(n == 1) return arr[0];
    int val = mystery(arr + 1, n - 1)
    return (arr[0] < val) ? arr[0] : val;
}
```

pointer arithmetic. makes next function have array minus first element

ternary operator

shorthand for

```
if(arr[0] <val)
    return arr[0];
else
    return val;
```




A. find the minimum element of arr B. find the maximum element of arr C. find the the sum of all elements of arr D. sort all elements of arr

1. A

2. [5 points] Which of the following descriptions best describes what **mystery** does?

```
bool mystery(int n, int i) {
    if (n <= 2)
        return (n == 2) ? true : false;
    if (n % i == 0)
        return false;
    if (i * i > n)
```

← If n is evenly divisible by i

← If $i^2 > n$

```
        return true;
    return mystery(n, i + 1);
}
```

← n unchanged
i increments by 1

mystery(6,2) = False
mystery(7,2) = True
mystery(8,2) = False
mystery(9,2) = False
mystery(11,2) = True
why pass in 2?
smallest # we can mod by, if i increases

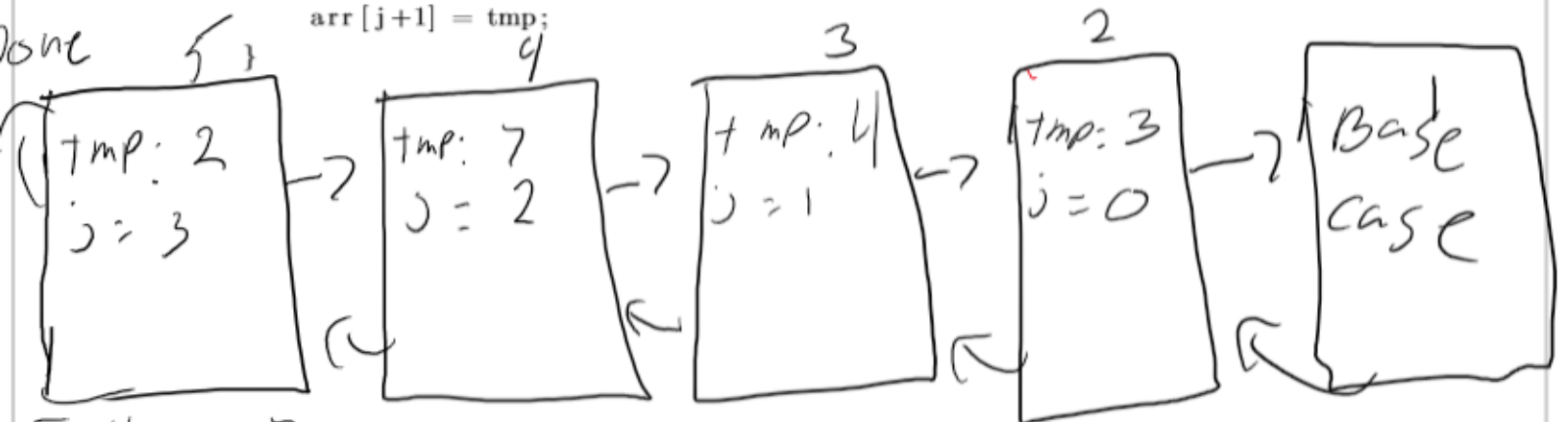
A. determine if n is an even number B. determine if n is a prime number C. determine if i evenly divides n D. determine if n is an odd number

2. B

3. [5 points] Given the following sorting algorithm, determine if it is **stable**, **in-place**, **both**, or **neither**.

```
int sort(int *arr, int n) {
    if (n <= 1) return;
    sort(arr, n-1);
    int tmp = arr[n-1];
    int j = n-2;
    while (j >= 0 && arr[j] > tmp) {
        arr[j+1] = arr[j];
        j--;
    }
    arr[j+1] = tmp;
}
```

[5,3,4,7,2]




[3,4,5,7,7]
[3,4,5,5,7]
[3,4,4,5,7]
No Changes
[3,5,5,7,2]
[3,4,5,7,2]
[5,5,4,7,2]
[3,5,4,7,2]

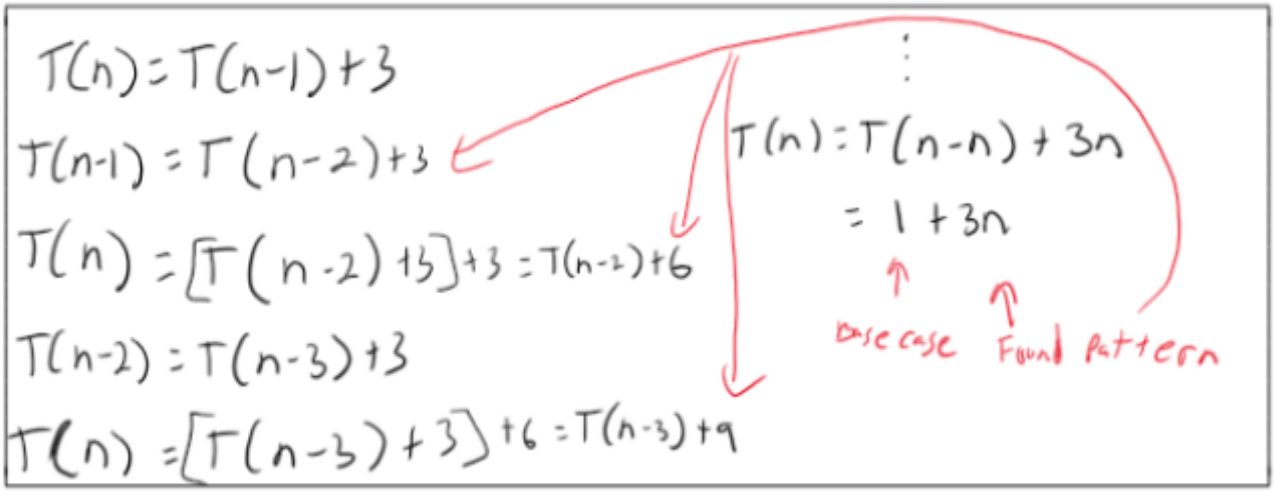
[3, 3, 4, 5, 7] -> [2, 3, 4, 5, 7]



A. stable B. in-place C. both D. neither

3. C

4. [10 points] Solve the following recurrence relation: $T(0) = 1; T(n) = T(n-1) + 3$




A. $3n+1$ B. $3n-1$ C. $3n$

4. A

5. [10 points] Solve the following recurrence relation: $T(1) = 1; T(n) = 2T(n/2) + n$

$T(n) = 2T(\frac{n}{2}) + n$

$T(\frac{n}{2}) = 2T(\frac{n}{4}) + \frac{n}{2}$

$T(n) = 2T\left(2T(\frac{n}{4}) + \frac{n}{2}\right) + n$

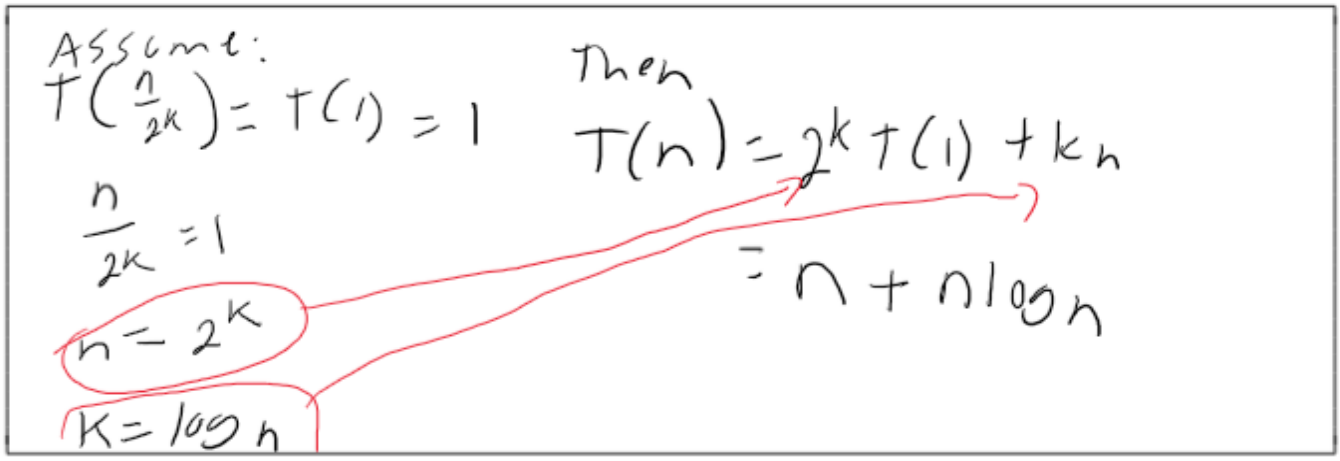
$= 2^2 T(\frac{n}{2^2}) + n + n$

$T(\frac{n}{2^2}) = 2T(\frac{n}{2^3}) + \frac{n}{2^2}$

$T(n) = 2^2 T\left(2T(\frac{n}{2^3}) + \frac{n}{2^2}\right) + 2n$

$= 2^3 T(\frac{n}{2^3}) + 3n$

$= 2^k T(\frac{n}{2^k}) + kn$




A. $n + logn$ B. $nlogn$ C. $n + nlogn$ D. $n^2 + nlogn$

5. C

6. [5 points] Is a linked list the best underlying structure to implement a queue with? Justify your answer.

Yes. Queues have two primary operations: enqueue and dequeue.
LinkedList w/ a tail pointer gives us O(1) cost for both operations, and we can grow to any size with no additional cost.

A. Yes B. No

6. A

7. Would a stack or queue be more efficient for the following:

(a) [3 points] An undo button in a text editor

Each action gets pushed onto the stack. undo is simply popping actions off

(a) stack

(b) [3 points] A web server

Handle requests in the order they arrive

(b) queue

(c) [3 points] A breadth-first search

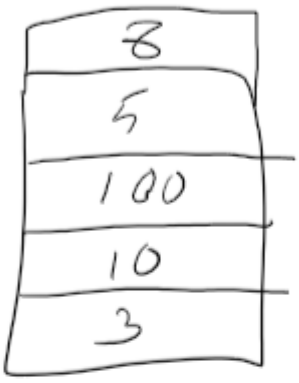
store the nodes to visit in FIFO order (c) queue

(d) [3 points] A depth-first search

store the nodes to visit in FILO order (d) stack

8. [5 points] Given the following function **mystery**, determine its output assuming **stack** has had the following elements inserted in order: 3, 10, 100, 5, 8

```
int mystery(std::stack<int> stack) {
    int result = 0;
    int loop = stack.size(); 5
    for(int i = 0 ; i < loop; i++) {
        if(!(i % 2)) {
            result += stack.top();
        }
        else {
            result *= stack.top();
        }
        stack.pop();
    }
    return result;
}
```




0 % 2 = 0
!0 = 1
result += 8 = 8

1 % 2 = 1
!1 = 0
result *= 5 = 40

2 % 2 = 0
!0 = 1
result += 100 = 140

3 % 2 = 1
!1 = 0
result *= 16 = 1,400

4 % 2 = 0
!0 = 1
result += 3 = 1,403

A. 1403 B. 658 C. 1530 D. 8040

8. A