(2) [5+10=15 points]
Here is the code for a base-b counter. (For this entire problem, you get to assume that $b \geq 2$.)

```
class Counter {
  private:
    int n;
    int b;
    int *p;
  public:
    Counter (int b, int n) {
       this->n = n;  this->b = b;
       p = new int [n];
       for (int i = 0; i < n; i ++) p[i] = 0;
    }
    void increment () {
       int i;
       for (i = 0; i < n && p[i] == b-1; i ++)
           p[i] = 0;
       p[i] ++;
   }
}
int main() {
  int n;
  cin >> n;
  Counter c(2, n);
  for (int i = 0; i < pow(2, n); i++) c.increment();
  return 0;
}
```

(a) Analyze the **worst**-case runtime of `increment()` in terms of `n` using Θ-notation, and explain your answer.

If p[i] == b-1 for all i from 0 to n-1
then loop runs n times, so $\Theta(n)$.

(b) Analyze the **worst**-case runtime of `main()` in terms of `n` using Θ-notation, and explain your answer. **Hint:** figure out how much work is spent changing p[0], p[1], etc, and sum these values together.

p[0] increments half the runs ($2^n/2$) and decrements for the other half ($2^n/2$). ($W[0] = 2^n$)

p[i] increments when p[i-1] decrements (half the time) and decrements the next round. ($W[i] = W[i-1]/2$)

$$T(n) = \sum_{i=0}^{n-1} W[i] = \sum_{i=0}^{n-1} 2^n/2^i = \Theta\left(\sum_{i=0}^{n} 2^i\right) = \Theta\left(\frac{1-2^{n+1}}{1-2}\right)$$

$$= 2^{n+1} - 1 = \Theta(2^n)$$

(3) [5+10=15 points]
Brutus the Bruin has implemented the following lazy version of Quicksort:

```
void Quicksort (T a[], int l, int r) {
   if (l < r) {
     int m = median-partition(a,l,r);
     Quicksort (a, l, m-1);
   }
}
```

`median-partition(a,l,r)` runs in linear time $\Theta(r-l)$ and finds the *actual median* of the array between l and r, then uses it as a pivot to partition the array between l and r. Notice that Brutus unfortunately only recursively sorts the left subarray, and is forgetting about the right part. So the algorithm isn't particularly useful. Nonetheless, you will analyze here how long it takes.

(a) Set up a recurrence relation for the running time, and describe where the terms came from.

$T(i) = T(i/2) + \Theta(i)$

$\Theta(i)$ from running median-partition

$T(i/2)$ from recursive call, since $(\frac{i}{2})$ terms are less than the median.

(b) Derive a solution to the recurrence relation using a method of your choice. Show your work.

$$T(n) = T(\tfrac{n}{2}) + \Theta(n)$$
$$= T(\tfrac{n}{4}) + \Theta(n) + \Theta(\tfrac{n}{2})$$
$$= T(\tfrac{n}{8}) + \Theta(n) + \Theta(\tfrac{n}{2}) + \Theta(\tfrac{n}{4})$$
$$\vdots$$
$$= \Theta(n) + \Theta(\tfrac{n}{2}) + \Theta(\tfrac{n}{4}) + \cdots + \Theta(1)$$
$$= \sum_{i=0}^{\log(n)} \Theta(2^i) = \Theta\left(2 \sum_{i=1}^{\log(n)-1} 2^i\right)$$
$$= \Theta\left(2\,\frac{(1-2^{\log(n)})}{(1-2)}\right) = \Theta(2(n-1))$$
$$= \Theta(n)$$