Name: Into Chot **ID:** 1801087.

Problem 5: Big-O

For all of the following, determine the **total operation count function** and then the **Big-O** of the given code segments. Remember that you are mostly just counting the number of times something happens, if it helps you can plug in some numbers to get a sense of how many times certain things occur, and then generalize from that.

a.

```
for (int i = 0; i < p; i++)
        sum++;
```

1. Int i = 0 — 1
2. i < p — 1 * p = p
3. i++ — 1 * p = p
4. sum++ — 1 * p = p

⇒ f(p) = p + p + p + 1 = 3p + 1

⇒ O(p).

b.

```
int k = m;
while (k > 0) {
        sum++;
        k--;
}
```

m = 3

k = 3.

sum	k
0	3
1	2
2	1

⇒ f(m) = 1 + m + m + m = 3m + 1

⇒ O(m)

c.

```
for (int j = 0; j < z; j++) {
        int i = 0;
        while (i < z) {
                sum++;
                i++;
        }
}
```

1. Int j = 0 — 1
2. j < z — 1 × z = z
3. j++ — 1 × z = z.
4. Int i = 0 — 1 * z = z.
5. while — z * z = z^2.
6. sum++ — z^2
7. i++ — z^2

⇒ f(n) = $3z^2 + 3z + 1$

⇒ $O(z^2)$.

Name: Inho Choi **ID:** 1801[illegible]

For all the following just provide the Big-O:

d.

```
int magicfact(int n) {
        int mult = 1;
        for (int i = 1; i <= n; i++)
                mult *= i;
        return mult;
}
```

In this case, Big-O is $O(n)$.

e.

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

In this case, Big-O is $O(n)$.

n=5.
5 * fact(4) 120.
4 * fact(3) 24
3 * fact(2) 6
2 * fact(1) 2
1 * fact(0) 1

f.

```
for (int i = 0; i < q*q; i++)
        for (int j = 0; j < i; j++)
                sum++;
```

In this case, Big-O is (q^4)

1. int i = 0 — 1
2. i < q*q — q^2
3. i++ — q^2
4. int j = 0 — $1 \cdot q^2$
5. j < i — $1 \cdot \frac{q^2 \cdot (q^2-1)}{2}$
6. j++ — $\frac{q^2 \cdot (q^2-1)}{2} \cdot 1$
7. sum++ — $\frac{q^2 \cdot (q^2-1)}{2} \cdot 1$

$f(n) = \frac{3}{2}(q^2 \cdot (q^2-1)) + 3q^2 + 1$.
$O(q^4)$

$(n-1)^2 - 1$

g.

```
for (int i = 0; i < n; i++)
        for (int j = 0; j < i*i; j++)
                for (int k = 0; k < j; k++)
                        sum++;
```

In this case, Big-O is $O(n^5)$

1. int i = 0 — 1
2. i < n — n
3. i++ — n
4. int j = 0 — n
5. j < i++ — $\frac{(n-1)(n-1)}{2}$
6. j++ — $\frac{(n-1)(n-1)}{2}$
7. int k = 0. — $n * \frac{(n-1)(n-1)}{2}$
8. k < j — $n \times \frac{(n-1)(n-1)}{2} * \frac{(n-1)^2-1}{2}$
9. k++ — $n \times \frac{(n-1)(n-1)}{2} * \frac{(n-1)^2-1}{2}$
10. sum++ — $n \times \frac{(n-1)(n-1)}{2} * \frac{(n-1)^2-1}{2}$

Name: Inho Choi **ID:** 1801[illegible].

h.

```
for (int i = 0; i < p; i++)
      for (int j = 0; j < i*i; j++)
            for (int k = 0; k < i; k++)
                  sum++;
```

1. P
2. $\frac{(P-1)(P-1)}{2}$.
3. $P \cdot \frac{(P-1)^2}{2} \cdot (P-1)$.

In this case, Big-O is $O(P^4)$.

i.

```
for (int i = 0; i < n; i++)        n.
{
      Circ arr[n];                 n
      arr[i].setRadius(i);         constant.
}
```

In this case, Big-O is $O(n)$

j.

```
for (int i = 0; i < n; i++)        n.
{
      int k = i;                   n.
      while (k > 1)
      {
            sum++;                 1 1 2 2 3.
            k = k / 2;
      }
}
```

In this case, Big-O is $O(n \cdot \log_2 n)$.

Name: Ino Choi **ID:** 180178N.

Problem 2:

Given a vector of sets of ints, `vector< set<int> > v`, assume the vector v has **N** total sets and that each set has an average of **Q** items.

a. What is the Big-O of determining if the first set, v[0], contains the value 7?

$O(\log_2 Q)$

b. What is the Big-O of determining if any set in v has the value 7?

$O(n * \log_2 Q)$

c. What is the Big-O of determining the number of even values in all of v?

$2N * Q$

$O(N*Q)$

d. What is the Big-O of finding the first set with a value of 7 and then counting the number of even values in that set?

$O(n \log_2 Q)$ + $2N * Q$

$\Rightarrow O(n \log_2 Q + 2N*Q)$.