Neil Opera 110878452

a) sumPairs program

```
public static boolean sumPairs(int[] arr, int key) throws
IllegalArgumentException{
   if(arr.length < 1)
       throw new IllegalArgumentException("Array size invalid");
   ArrayList<Value> pairs = new ArrayList<>();
   for (int i = 0; i < arr.length; i++)
       for (int j = i + 1; j < arr.length; j++) {
           if (arr[i] + arr[j] == key) {
               Value temp = new Value(arr[i], arr[j]);
               pairs.add(temp);
           }
       }
   }
   if (pairs.size() > 0) {
       for (int i = 0; i < pairs.size(); i++) {
           System.out.println(pairs.get(i).showPairs());
       }
       return true;
   }
   return false;
}
```

$n^2+1-1-1 = n^2-1$

max size of pairs.size() $= (n-1)+(n-2)+(n-3)+\ldots+1$

$= \frac{n^2}{2}$

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

$O(n) = n^2$

b) sumTriplets program

```
public static boolean sumTriplets(int[] arr, int key) throws
IllegalArgumentException{
   if(arr.length < 1)
       throw new IllegalArgumentException("Array size is invalid");
   ArrayList<Value> triplets = new ArrayList<>();
   for (int i = 0; i < arr.length; i++)
       for (int j = i + 1; j < arr.length; j++) {
           for (int k = j + 1; k < arr.length; k++) {
               if (arr[i] + arr[j] + arr[k] == key) {
                   Value temp = new Value(arr[i], arr[j], arr[k]);
                   triplets.add(temp);
               }
```

$n^3+1-1-2 = n^3-2$

```
            }
        }
    }
    if (triplets.size() > 0) {
        for (int i = 0; i < triplets.size(); i++) {
            System.out.println(triplets.get(i).showTriplets());
        }
        return true;
    }
    return false;
}
```

→ 1

→ $\frac{n(n-1)}{2}+1$

→ $\frac{n(n-1)}{2}$

← 1

max size of triplets.size():
$(n-2)+(n-3)+(n-4)\dots+1$
$\frac{n(n-1)}{2}$

$T(n) = 1+1+n+1+n^2-1+n^3-2$
$+n^3-3+n^3-3+n^3-3+1+1+\frac{n(n-1)}{2}+\frac{n(n-1)}{2}+1$

$O(n) = n^3$

b) matrix multiplication

```
for(int i = 0; i < n; i++){
    for(int j = 0; j < n; j++){
        double sum = 0;
        for(int k = 0; k < n; k++){
            sum += a[i][k] * b[k][j];
        }
        C[i][j] = sum;
    }
}
```

→ $n+1$

→ $n^2+1-1 = n^2$

→ n^2-1

→ $n^3-2+1 = n^3-1$

↳ n^3-2

↳ n^2-1

$T(n) = n+1+n^2+n^2-1+n^3-1$
$+n^3-2+n^2-1$

$O(n) = n^3$