Algorithm Analysis Worksheet

Name:_____

1. Give the **best** and **worst** case running times (using Θ notation) for each of the following in terms of the size of the input.

   (a)
```
void foo(int n) {
    int foo = 0;
    for(int i = 0 ; i < n ; i++)
      foo += i;
}
```

   (b)
```
void blah(int n) {
    int blah = 0;
    for(int i = 0 ; i < sqrt(n) ; i++)
      blah += i;
}
```

   (c)
```
void ferzle(int a[], int n) {
    int ferzle = 0;
    for(int i = 0 ; i < n ; i++) {
        for(int j = 0 ; j < n ; j++) {
            ferzle += a[i]*a[j];
            if(ferzle==10000) {
                j=n;
            }
        }
    }
}
```

   (d)
```
void ferzle2(int n) {
    int ferzle = 0;
    for(int i = 0 ; i < n ; i++) {
        for(int j = i ; j < n ; j++) {
            ferzle += i*j;
        }
    }
}
```

(e)
```
void ferzle3(int a[], int n) {
    int ferzle = 0;
    for(int i = 0 ; i < n ; i++) {
        for(int j = 0 ; j < n ; j++) {
            ferzle += a[i]*a[j];
            if(ferzle==10000) {
                i=n;
            }
        }
    }
}
```

(f)
```
void ferzle4(int a[], int n) {
    int ferzle = 0;
    for(int i = 0 ; i < n ; i++) {
        for(int j = 0 ; j < n ; j++) {
            ferzle += a[i]*a[j];
        }
        if(ferzle==10000) {
            i=n;
        }
    }
}
```

(g)
```
void gruhop1(int n) {
    int gruhop = 0;
    for(int i = 0 ; i < n/2 ; i++)
        for(int j = 0 ; j < n/2 ; j++)
            gruhop += i*j;
}
```

(h)
```
void gruhop2(int n) {
    int gruhop = 0;
    for(int i = 0 ; i < sqrt(n) ; i++)
        for(int j = 0 ; j < n ; j++)
            gruhop += i*j;
}
```

(i)
```
int sumSomeStuff(int []A) {
    int sum=0;
    int i=0;
    while(i < A.length) {
        sum = sum + A[i];
        i++;
        if(sum > 100000) {
            i=A.length;
        }
    }
    return sum;
}
```

(j)
```
int doMoreStuff(int []A) {
    int sum=0;
    for(int i=0 ; i < A.length ; i++) {
        for(int j=0 ; j < A.length ; j++) {
            sum = sum + A[i]*A[j];
        }
        if(sum==123) {
            i = A.length;
        }
    }
    return sum;
}
```

(k)
```
int sumTimesM(int []A) {
    int M = 100;
    int sum=0;
    for(int i=0 ; i < A.length ; i++) {
        for(int j=0 ; j < M ; j++) {
            sum = sum + A[j] + A[i];
            if(sum==123) {
                j = M;
            }
        }
    }
    return sum;
}
```

(l)
```
void foo(int n,int m) {
    int foo = 0;
    for(int i = 0 ; i < n ; i++)
        foo++;
    for(int j = 0 ; j < m ; j++)
        foo++;
}
```

(m)
```
Matrix MatrixMultiply(Matrix A, Matrix B) {
    Matrix C;
    for(int i=0 ; i < n; i++) {
        for(int j=0 ; j < n ; j++) {
            C[i][j]=0;
            for(int k=0 ; k < n ; k++) {
                C[i][j] += A[i][k]*B[k][j];
            }
        }
    }
    return C;
}
```

(n)
```
void foo2(int n) { // Tricky one
    int foo = 0;
    for(int i = 0 ; sqrt(i)  <  n ; i++)
        for(int j = 0 ; j < i; j++)
            doIt(j) // takes time O(j);
}
```

(o)
```
void HalfIt(int n) {
    while(n > 0) {
        n = n/2;
    }
}
```