Homework 2

Stefan Dierauf

1.

```
{pre: items.size > 0}

k = 0
max = items[k]

{inv: max holds largest value in items[0..k] <- new invariant}
while(k != items.size) {
    k = k+1 //invariant broken
    if (max < items[k]) {
        max = items[k] //invariant restored
    }
}
```

Have to break the invariant by updating k before max is calculated, then invariant holds again after max is calculated again

2.

```
function int expt(int x, int y) {
{pre: x = x_0 & y = y_0 & x >= 0 & y >= 0}
z = 1

{inv: z * x^(y) = z_pre * x_pre ^ (y_pre) where _pre denotes that value at it's
previous iteration}

while (y > 0) {
    if (y % 2 == 0) {
        y = y/2
        x = x * x
    } else {
        z = z * x
        y = y-1
    }
}

{post: y = 0 so x^0 = 1 so z * x^(y) = z = x_0 ^ (y_0)}
```

3.

```
{pre: arr is an int[]}
int a = arr[0]
int i = 1;
int j = arr.length - 1;

{inv: arr & arr[1..i-1] <= a & arr[j..arr.length-1] > a}
while (i < j) {
    if (i > a) {
        swap(arr, i , j)
        j--;
    } else {
        i++;
    }
}
swap(arr, 0, i-1);

{post: arr[0..i-1] <= a & arr[i-1] = a & i = j & arr[j..arr.length - 1] > a}
```

4.

```
{pre: a is int[]}

{inv: a[0..i-1] is sorted lowest to highest}

for (int i = 0; i < a.length; i++) {
    int lowest = i;

    {inv: lowest = index of lowest number in a[i..j-1]}
    for (int j = i+1; j < a.length; j++) {
        if (a[j] < a[lowest]) {
            lowest = j;
        }
    }

    swap(a, i, lowest);
}

{post:i = a.length so a[0..a.length-1] is sorted from lest to greatest }
```