# Homework 2

* Problem 1

1. The function method1 contains only one loop and it is executed based on the value of “n”.

public int method1(int[] a) {

int x = 0; // c1

int y = 0; // c2

for (int i=1; i<n; i++) { // loop executed n-1 times

if (a[i] == a[i-1]) { // loop body takes c3

x = x + 1;

} else {

y = y + 1;

}

}

return (x - y); // c4

}

Total running time, f(n) = c1 +c2 + c3(n-1) + c4

F(n) = o(n)

1. The function method2 contains 3 loops and it is executed based on the value of “n”

public int method2(int[] a, int[] b) {

int x=0; // c1

int i=0; // c2

while (i < n) { // loop executed n times

y=0;

j=0;

while (j<n){ // loop executed n times

k=0;

while(k<=j){ // loop executed (n+1)/2 times

y=y+a[k]; // loop body takes c3

k=k+1;

}

j=j+1;

}

if (b[i]==y){

x++;

}

i = i+1;

}

return x; // c4

}

Total running time, f(n) = c1 + c2 + c3(n2(n+1)/2) + c4

F(n)=O(n3)

1. The function method3 recursive for n-1 times

public void method3(int[] a, int i, int[] p) {

if (i == 0) { // c1

p[0] = a[0];

p[1] = a[0];

} else { // c2

method3(a, i-1, p); // recursive a.length - 1 times

if (a[i] < p[0]) { // c3

p[0] = a[i];

}

if (a[i] > p[i]) { // c4

p[1] = a[i];

}

}

}

F(n) = c1 + c2(n-1) + c3 + c4 = O(n)

1. The function method4 recursive for (n-1)/2 times

public static int method4(int[] a, int x, int y) {

if (x >= y) { // c1

return a[x];

} else {

int z = (x + y) / 2; // c2

int u = method4(a, x, z); // c3, recursive (n-1)/2 times

int v = method4(a, z+1, y); // c4 recursive (n-1)/2 times

if (u < v) {. // c5

return u;

} else { // c6

return v;

}

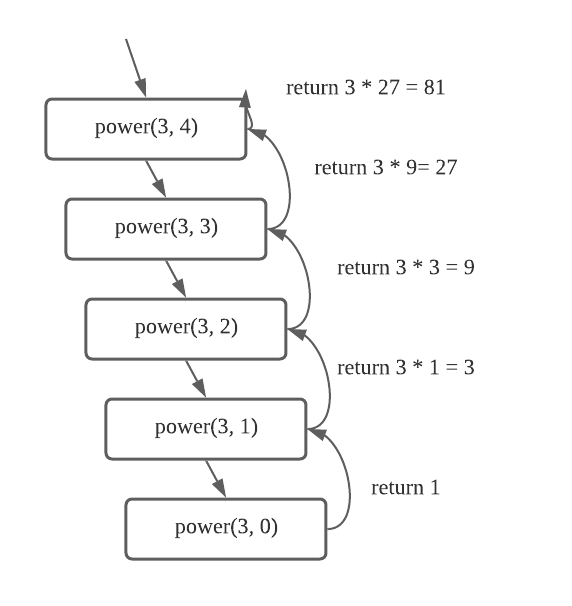
}

}

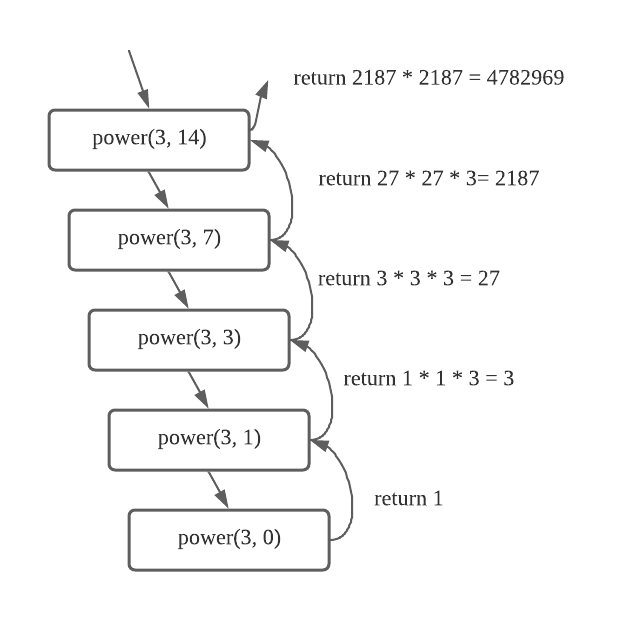
F(n) = c1 + c2 + c3(logn) + c4(logn) + c5 + c6 = O(logn)

* Problem 2

1. Recursion trace



1. Recursion trace



* Problem 3

1. table

|  |  |  |
| --- | --- | --- |
| Operation | Return Value | Stack Contents |
| Push(10) | - | (10) |
| Pop() | 10 | () |
| Push(12) | - | (12) |
| Push(20) | - | (12, 20) |
| Size() | 2 | (12, 20) |
| Push(7) | - | (12, 20, 7) |
| Pop() | 7 | (12, 20) |
| Top() | 20 | (12, 20) |
| Pop() | 20 | (12) |
| Pop() | 12 | () |
| Push(35) | - | (35) |
| isEmpty() | false | (35) |

1. table

|  |  |  |
| --- | --- | --- |
| Operation | Return Value | Queue Contents (first <-Q<-last) |
| Enqueue(7) | - | (7) |
| Dequeue() | 7 | () |
| Enqueue(15) | - | (15) |
| Enqueue(3) | - | (15, 3) |
| First() | 15 | (15, 3) |
| Dequeue() | 15 | (3) |
| Dequeue() | 3 | () |
| First() | null | () |
| Enqueue(11) | - | (11) |
| Dequeue() | 11 | () |
| isEmpty() | true | () |
| Enqueue(5) | - | (5) |