Problems Solved:

46	47	48	49	50

Name:

Matrikel-Nr.:

Problem 46. Let $T(n)$ be the total number of times that the instruction `a[i,j] = a[i,j] + 1` is executed during the execution of `P(n,0,0)`.

```
procedure P(n, x, y)
    if n >= 1 then
        for (i = x; i < x+n; i++)
            for (j = y; j < y+n; j++)
                a[i,j] = a[i,j] + 1
        h = floor( n / 2)
        P(h, x,   y  )
        P(h, x+h, y  )
        P(h, x,   y+h)
        P(h, x+h, y+h)
    end if
end procedure
```

1. Compute $T(1)$, $T(2)$ and $T(4)$.
2. Give a recurrence relation for $T(n)$.
3. Solve your recurrence relation for $T(n)$ in the special case where $n = 2^m$ is a power of two, i.e. derive a guess for and explicit expression of $T(2^m)$ and then prove this formula by induction.
4. Use the Master Theorem to determine asymptotic bounds for $T(n)$.

Problem 47. Let $T(n)$ be the number of multiplications carried out by the following Java program.

```
int a, b, i, product, max;
max = 1;
a = 0;
while ( a < n ) {
  b = a;
  while (b <= n) {
    product = 1;
    i = a;
    while (i < b) {
      product = product * factors[i];
      i = i + 1; }
    if (product > max) { max = product; }
    b = b + 1; }
  a = a + 1; }
```

1. Determine $T(n)$ exactly as a nested sum.

2. Determine $T(n)$ asymptotically using Θ-Notation by a derivation that justifies your result. In your derivation, you may use the asymptotic equation

$$\sum_{k=0}^{n} k^m = \Theta(n^{m+1}) \text{ for } n \to \infty$$

for fixed $m \geq 0$ which follows from approximating the sum by an integral:

$$\sum_{k=0}^{n} k^m \simeq \int_0^n x^m \, dx = \frac{1}{m+1} n^{m+1} = \Theta(n^{m+1})$$

Problem 48. Consider the following pseudo code of an implementation of a FIFO (first in first out) queue with two functions enqueue and dequeue.

```
input  := EMPTYLIST
output := EMPTYLIST
function enqueue(e, input, output) { push(e, input) }
function dequeue(input, output) {
    if isempty(output) {
        while not isempty(input) { push(pop(input), output) }
    }
    pop(output)
}
```

Analyze its amortized cost of these functions by (a) the aggregate method and (b) the potential method.
Here,

- `push(e, L)` is the operation of adding an element e to the front of a list L,
- `isempty(L)` returns `TRUE` if the list L is empty,
- `pop(L)` is the operation that removes the first element of a list L and returns it.

All these operations are assumed to cost constant time.
In the code above, a queue is represented by a pair (`input`, `output`). Putting a new element into the queue via `enqueue`, first puts it to the front of `input`. Only when an element is requested via a call to `dequeue`, elements are moved from `input` to `output` list, thus effectively reversing `input` so that in total the queue returns its elements in a FIFO principle.
Hint: For the potential method you might want to consider the function Φ such that for a queue q that is represented by the pair (`input`, `output`) of two lists, $\Phi(q)$ is the size of the `input` list.

Problem 49. Consider a RAM program that evaluates the value of $\sum_{i=1}^{n} i^2$ in the naive way (by iteration). Analyze the worst-case asymptotic time and space complexity of this program assuming the existence of operations `ADD r` and `MUL r` for the addition and multiplication of the accumulator with the content of register r.

1. Determine a Θ-expression for the number $S(n)$ of registers used in the program with input n (space complexity).

2. Determine a Θ-expression for the number $T(n)$ of instructions executed for input n (time complexity in constant cost model),

3. Assume a simplified version of the logarithmic cost model of a RAM where the cost of every operaton is proportional to the length of the arguments involved. In particular, if a is the (bit) length of the accumulator and l is the (bit) length of the content of register r then `MUL r` costs $a+l$ and `ADD r` costs $\max(a, l)$.

   Determine the asymptotic costs $C(n)$ (using O-notation) of the program for input n.

As usual, you must give appropriate justification for each of your results.

Problem 50. Take the following recursive program.

```
1 f(n,b) ==
2     if n < 1 then return 0
3     d := floor(n/3)
4     return b + f(d,1) + 2*f(d,2)
```

Let $C(n)$ be the number of comparisons executed in line 2 while running $f(n, 0)$ for some positive integer n.

1. Write down a recurrence for C and determine enough initial values.

2. Solve that recurrence for the given initial values and arguments n of the form $n = 3^m$, i.e., derive a guess for a closed form expression for $C(3^m)$.

3. Prove by induction that your guess is correct.