Algorithmen & Datenstrukturen
Wiederholungsfragen Schemas

1. Aufgabe: Analysieren Sie folgenden Algorithmus auf Korrektheit. Bringen Sie falls nötig Korrekturen an.

```
public static int Factorial(int n) {
    Abbruchbedingung fehlt
    return n * Factorial(n - 1);

}
```

2. Aufgabe: In welcher Komplexitätsklasse befindet sich die rekursive Variante des Fibonacci-Algorithmus? Wieso befindet sich die iterative Variante in einer anderen Komplexitätsklasse (welcher?)?

Der rekursive algorythmus ruft immer zwei mal sich selbst auf, somit O(2^n)
Der iterative läuft für jede zahl nur einmal. somit O(n)

3. Aufgabe: Um das Maximum einer Menge S mit n Elementen zu bestimmen, kann das folgende Divide-and-Conquer Verfahren angewendet werden:

1. Divide: Zerlege die Menge S in zwei Teilmengen S_1 und S_2 mit $|S_1| = \left\lceil \frac{|S|}{2} \right\rceil$ und $|S_2| = \left\lfloor \frac{|S|}{2} \right\rfloor$.
2. Conquer: Wende das Verfahren rekursiv an, um $\max(S_1)$ und $\max(S_2)$ zu berechnen.
3. Combine: Berechne daraus $\max(S)$. Überlegen Sie sich den trivialen Fall, der dann die Berechnung durchführt und die Rekursion abbricht.

Formulieren Sie zu diesem Verfahren einen rekursiven Algorithmus in Pseudocode.

Hinweise:
$|S_1|$ = Anzahl Elemente der Menge S_1
$\lfloor x \rfloor$ = x auf die nächst kleinere Ganzzahl abrunden (C#: `Math.Floor()`)
$\lceil x \rceil$ = x auf die nächst grössere Ganzzahl aufrunden (C#: `Math.Ceiling()`)

4. Aufgabe: Gegeben ist folgendes Programm:

```
1       public class Recursion {
2           public static int Rec(int p1, int p2) {
3               if (p2 == 0) {
4                   return 1;
5               }

6               if ((p2 % 2) != 0) {
7                   int y = Rec(p1, (p2 - 1) / 2);
8                   Console.WriteLine($"{p1} * {y} * {y}");
9                   return p1 * y * y;
10              } else {
11                  int y = Rec(p1, p2 / 2);
12                  Console.WriteLine($"{y} * {y}");
13                  return y * y;
14              }
15          }
16
17          public static void Main() {
18              Rec(2, 5);
19          }
20      }
```

Auf den beiden Zeilen 9 und 13 werden jeweils Multiplikationen durchgeführt. Zusätzlich werden die Multiplikationen auf der Konsole ausgegeben.
In nachfolgender Tabelle sollen in zeitlicher Reihenfolge die Ausgaben des Programms (d.h. die Multiplikationen in zeitlicher Abfolge) notiert werden (z.B. «2 * 3 * 4»), für den Fall, dass `Main()` ausgeführt wird.

1. Multiplikation:	2 * 1 * 1
2. Multiplikation:	2 * 2
…	2 * 4 * 4

5. Aufgabe: Erklären Sie das Schema "Divide and Conquer" mit eigenen Worten. Kennen Sie ein Beispiel?

Man suchte etwas indem man das suchfeld immer teilt (bsp halbiert), suche einer Zahl in einem sortierten Array

6. Aufgabe: Erklären Sie das Schema "Greedy" mit eigenen Worten. Kennen Sie ein Beispiel?

Man nimmt immer den besten schritt mit den infromationen die man kennt.