

Aufgabe 1:

Rekursions-Experimente

Prompt:

was macht dieses Java Programm?

```
public static void recursive(int depth){
    System.out.println(depth);
    recursive(depth+1);
}
public static void recursive2(int depth){
    int[] array = new int[100];
    System.out.println(depth);
    recursive2(depth+1);
}
public static void recursive3(int depth){
    int var0=3;int var1=3;int var2=3;int var3=3;int var4=3;int var5=3;int
var6=3;int var7=3;int var8=3;int var9=3;int var10=3;
    int var11=3;int var12=3;int var13=3;int var14=3;int var15=3;int
var16=3;int var17=3;int var18=3;int var19=3;int var20=3;
    int var21=3;int var22=3;int var23=3;int var24=3;int var25=3;int
var26=3;int var27=3;int var28=3;int var29=3;int var30=3;
    int var31=3;int var32=3;int var33=3;int var34=3;int var35=3;int
var36=3;int var37=3;int var38=3;int var39=3;int var40=3;
    int var41=3;int var42=3;int var43=3;int var44=3;int var45=3;int
var46=3;int var47=3;int var48=3;int var49=3;int var50=3;
    int var51=3;int var52=3;int var53=3;int var54=3;int var55=3;int
var56=3;int var57=3;int var58=3;int var59=3;int var60=3;
    int var61=3;int var62=3;int var63=3;int var64=3;int var65=3;int
var66=3;int var67=3;int var68=3;int var69=3;int var70=3;
    int var71=3;int var72=3;int var73=3;int var74=3;int var75=3;int
var76=3;int var77=3;int var78=3;int var79=3;int var80=3;
    int var81=3;int var82=3;int var83=3;int var84=3;int var85=3;int
var86=3;int var87=3;int var88=3;int var89=3;int var90=3;
    int var91=3;int var92=3;int var93=3;int var94=3;int var95=3;int
var96=3;int var97=3;int var98=3;int var99=3;
    System.out.println(depth);
    recursive3(depth+1);
}
}
```

Antwort:

Dieses Java-Programm definiert drei Methoden: recursive, recursive2, und recursive3. Jede Methode ruft sich selbst rekursiv auf. Die Methode recursive gibt einfach den übergebenen Tiefenwert aus und ruft sich dann mit einer um eins erhöhten Tiefe auf. Die Methode recursive2 erstellt außerdem ein Array mit 100 Elementen und gibt den Tiefenwert aus, bevor sie sich selbst aufruft. Die Methode recursive3 erstellt 100 Variablen und gibt dann den Tiefenwert aus, bevor sie sich selbst aufruft. Das endet nie und führt zu einer unendlichen Rekursion.

Prompt 2:

Was könnte der Sinn dahinter sein?

Antwort 2:

Es scheint, dass der Sinn hinter diesem Programm darin bestehen könnte, die Funktionsweise und das Verhalten von rekursiven Methoden zu untersuchen. Insbesondere soll möglicherweise das Verhalten von Methoden mit unterschiedlichen Variablen (einschließlich Arrays) und unterschiedlichen Datentypen in einer rekursiven Umgebung untersucht werden.

Allerdings ist es wichtig zu beachten, dass diese Art von unendlicher Rekursion in der Praxis vermieden werden sollte, da sie zu einem Stack Overflow führen kann. In realen Anwendungen ist es wichtig, Rekursion mit einer Abbruchbedingung zu versehen, um unendliche Schleifen zu vermeiden.

Bewertung:

Die Funktion wurde richtig erkannt und erklärt.

Nach erneuter Frage wurde auch ein guter möglicher Anwendungsfall geliefert der Grund sein könnte ein solches Programm zu schreiben.

Frage 2: Matrizen

Prompt:

Was kann dieses Programm?

```
public class matrix {
    int rows;
    int cols;
    int[][] field;

    matrix(int rows, int cols){
        this.rows = rows;
        this.cols = cols;

        field = new int[rows][cols];
    }

    void print(){
        for (int i=0; i<this.rows; i++){
            for (int j=0; j<this.cols; j++){
                System.out.print(field[i][j]);
                if (j != this.cols-1) {
                    System.out.print(" ");
                }
            }
            System.out.println();
        }
        System.out.println("=====");
    }

    void setValue(int row, int col, int value){field[row][col] = value;}

    int getValue(int row, int col){return field[row][col];}

    void scalarMultiply(int factor){
        for (int i=0; i<this.rows; i++) {
            for (int j = 0; j < this.cols; j++) {
                field[i][j] *= factor;
            }
        }
    }
}
```

```
        }
    }
}

void addMatrix(matrix matrix2){
    for (int i=0; i<this.rows; i++) {
        for (int j = 0; j < this.cols; j++) {
            field[i][j] += matrix2.field[i][j];
        }
    }
}

public static void main(String[] args){
    matrix nr1 = new matrix(2, 3);
    matrix nr2 = new matrix(2, 3);

    nr1.setValue(1, 1, 1);
    nr2.setValue(0, 0, 4);

    nr2.addMatrix(nr1);
    nr2.print();

    nr2.scalarMultiply(8);
    nr2.print();
}
}
```

Antwort:

Dieses Java-Programm definiert eine Klasse `matrix`, die eine 2D-Matrix repräsentiert und verschiedene Operationen mit Matrizen ermöglicht.

Die Klasse hat folgende Eigenschaften:

- `rows` und `cols` zur Speicherung der Anzahl von Zeilen und Spalten der Matrix
- `field`, ein 2D-Array zur Speicherung der Elemente der Matrix

Die Klasse definiert verschiedene Methoden:

- `matrix(int rows, int cols)`: Ein Konstruktor, der eine Matrix mit einer gegebenen Anzahl von Zeilen und Spalten erstellt und das `field`-Array entsprechend initialisiert.

Vorlesung Programmieren | Selbststudium | 14.05.2024 | Matti Bos

- `print()`: Eine Methode, die die Matrix auf der Konsole ausgibt.
- `setValue(int row, int col, int value)`: Eine Methode, um den Wert eines bestimmten Elements der Matrix zu setzen.
- `getValue(int row, int col)`: Eine Methode, um den Wert eines bestimmten Elements der Matrix abzurufen.
- `scalarMultiply(int factor)`: Eine Methode zur skalaren Multiplikation einer Matrix mit einem bestimmten Faktor.
- `addMatrix(matrix matrix2)`: Eine Methode zur Addition einer anderen Matrix zu dieser Matrix.

Die `main`-Methode erstellt zwei Matrizen `nr1` und `nr2`, setzt einige Werte in die Matrizen, addiert dann `nr1` zu `nr2`, gibt `nr2` aus, multipliziert `nr2` mit 8 und gibt das Ergebnis erneut aus.

Insgesamt ermöglicht dieses Programm die Erstellung, Manipulation und Anzeige von Matrizen und führt auch Operationen wie Addition und skalare Multiplikation von Matrizen durch.

Bewertung:

Auch hier ist die Antwort sehr gut. Die Funktionsweise der `Main` Methode wird erläutert, sowie auch die Funktionsweise aller einzelnen Methoden innerhalb der Klasse.

Ebenso gibt ChatGPT eine kleine Zusammenfassung in der ein potentieller Nutzen des Programms dargestellt wird.