

Aufgabenblatt 1

Algorithmen und Datenstrukturen

Lesen sie alle Aufgaben genau durch, bevor sie mit der Bearbeitung beginnen.

Legen sie für die Bearbeitung ein Textdokument an. In diesem werden ihre Ergebnisse gesammelt. Ihren Code brauchen sie nicht abgeben (siehe letzte Aufgabe), erstellen sollen sie ihn aber trotzdem. Für die Bearbeitung benötigen sie eine (beliebige) Chat-KI ihrer Wahl. Falls sie kein Account o.ä. haben, so können sie einfach den Chat auf <https://duck.ai> verwenden.

Aufgabe 1: Maximalwert ermitteln

- Legen sie ein neues IntelliJ-Projekt namens "Aufgabenblatt1" an.
- Legen sie ein Package "aufgabenblatt1" für eigene Klassen im Projekt an.
- Legen sie ein Package "util" für die Util-Klasse an.
- Kopieren sie die Klasse Util aus dem letzten Semester (im neuen Moodle-Kurs zu finden) in den Ordner des neuen Projekts und darin den util-Ordner des Package.
- In der "Util"-Klasse liegt bereits eine Methode mit der folgenden Signatur:

```
public int findMax(int[] array)
```

- Die Methode soll das Element eines Arrays zurückgeben, welches größer oder gleich den anderen Elementen des Arrays ist.
- Legen sie eine eigene Klasse "ArrayTester" die folgende main-Methode an:

```
public static final void main(String[] args) {  
    int num = 10;  
    int[] valueArray = new int[num];  
    Util util = new Util();  
    util.fillArrayRandom(valueArray, 100*num);  
    System.out.println(util.findMax(valueArray));  
}
```

Modifizieren sie die Methode findMax() nun so, dass es die Anzahl der Schleifendurchläufe statt des gefundenen Elements zurückgibt. Erhöhen sie nun den Wert von num auf 20, 30, 40,...,100. Protokollieren sie die Ergebnisse in einem Spreadsheet (z.B. Excel, Libreoffice Calc oder Google Spreadsheet) und lassen sie sich davon eine Kurvendarstellung generieren. Exportieren sie die Kurve z.B. aus Excel als .pdf, oder Bilddatei und fügen sie es in ihr Textdokument unter Aufgabe 1 ein.

Aufgabe 2: First Match

- In der Klasse liegt bereits die folgende Methode vor:

```
public int firstMatch(int[] array0, int[] array1)
```

- Die Methode gibt das erste Element des ersten Arrays zurückgeben, welches im zweiten Array vorkommt.
- Legen sie eine Klasse "MatchTester" im Package "aufgabenblatt1" mit der folgenden main-Methode an:

```
public static final void main(String[] args)
{
    Util util = new Util();
    int num = 10;
    int[] array0 = new int[num];
    int[] array1 = new int[num];
    util.fillArrayRandom(array0, num*100);
    util.fillArrayRandom(array1, num*100);
    System.out.println(util.firstMatch(array0, array1));
}
```

Modifizieren sie die Methode firstMatch() nun so, dass es die Anzahl der Schleifendurchläufe statt des gefundenen Elementes zurückgibt. Erhöhen sie nun den Wert von num auf 20, 30, 40,...,100. Protokollieren sie die Ergebnisse in einem Spreadsheet (z.B. Excel oder Google Spreadsheet) und lassen sie sich davon eine Kurvendarstellung generieren und fügen sie es in ihr Textdokument unter Aufgabe 2 ein.

Aufgabe 3: Zahlenraten

- Legen sie eine neue Klasse "NumberGuesser" an.
- Die Klasse soll eine Instanzvariable vom Typ int und dem Namen "number" enthalten.
- Schreiben sie für die Klasse einen Konstruktor mit der folgenden Signatur, welcher die Instanzvariable initialisiert:

```
public NumberGuesser(int targetnumber)
```

- Fügen sie der Klasse eine Methode mit der folgenden Signatur hinzu, welche zurückgibt, ob die übergebene Zahl "guess" größer als die Instanzvariable "number" ist:

```
public boolean isBigger(int guess)
```

- Fügen sie der Klasse eine Methode mit der folgenden Signatur hinzu, welche eine Zahl von der Konsole einliest und dann den Benutzer auf der Konsole informiert, ob die angegebene Zahl zu hoch, zu niedrig oder genau genau der Zielnummer entspricht:

```
public void guess()
```

- Fügen sie der Klasse eine Methode mit der folgenden Signatur hinzu, welche versucht, nur mit Hilfe der Methode "isBigger(int guess)" und dem direkten Vergleich mit == die Instanzvariable zu erraten:

```
public void autoGuess()
```

- **Überlegen sie dabei**, was eine gute Strategie sein könnte, probieren sie es ggf. Mit der vorigen Methode "von Hand" aus. Bei jedem Rateversuch soll die gerade geratene Zahl mit `System.out.println("Guess: " + guess);` ausgegeben werden. Testen sie ihre Methode.
- Versuchen sie herauszubekommen, was die Laufzeitkomplexität Ihres Ansatzes in asymptotischer Notation wäre. Falls noch nicht geschehen, versuchen sie einen Ansatz zu finden, der effizienter als $O(n)$ ist. Falls Laufzeitkomplexität noch nicht in der Vorlesung dran war, sparen sie es sich für später auf.
- Lassen sie sich von einer KI ihrer Wahl die Aufgabe, insbesondere die Methode `autoGuess()` generieren. Geben sie der KI dabei genug Informationen, um die Aufgabe bearbeiten zu können. Vergleichen sie ihre Lösung mit der KI Lösung. Wo liegen die Unterschiede? Was ist gleich? Schreiben sie die Antwort in ihr Textdokument unter Aufgabe 3 auf.
- Lassen sie die KI die Laufzeitkomplexität ihrer eigenen Lösung abschätzen. Liegt die KI richtig? Kopieren sie die KI-Antwort in das Textdokument und auch ihre Bewertung der Abschätzung.

Aufgabe 4: Anagramme

- Legen sie eine neue Klasse "AnagramChecker" an.
- Die Klasse soll zwei Instanzvariable vom Typ String und dem Namen "word0" und "word1" enthalten.
- Schreiben sie für die Klasse einen Konstruktor, welcher die Instanzvariable initialisiert.
- Fügen sie der Klasse eine Methode mit der folgenden Signatur hinzu, welche testet, ob word0 ein Anagramm von word1 ist:

```
public void isAnagram()
```

- Versuchen sie herauszubekommen, was die Laufzeitkomplexität Ihres Ansatzes in asymptotischer Notation wäre. Falls noch nicht geschehen, versuchen sie einen Ansatz zu finden, der effizienter als $O(n)$ ist. Falls Laufzeitkomplexität noch nicht in der Vorlesung dran war, sparen sie es sich für später auf.
- Lassen sie sich von einer KI ihrer Wahl die Aufgabe lösen. Geben sie der KI dabei genug Informationen, um die Aufgabe bearbeiten zu können. Vergleichen sie ihre Lösung mit der KI Lösung. Wo liegen die Unterschiede? Was ist gleich? Schreiben sie die Antwort in ihr Textdokument unter Aufgabe 4 auf.

Aufgabe 5: Primzahlen

- Legen sie eine neue Klasse "PrimeNumberGenerator" an.
- Die Klasse soll eine Instanzvariable vom Typ int und dem Namen "maximum" enthalten.
- Schreiben sie für die Klasse einen Konstruktor, welcher die Instanzvariable initialisiert.
- Fügen sie der Klasse eine Methode mit der folgenden Signatur hinzu, welche alle Primzahlen bis zum "maximum" auf der Konsole ausgibt. Legen sie dazu gerne eine Hilfsmethode an, z.B. einen Primzahltest. Überlegen sie sich eine geschickte Strategie, um die Methode möglichst effizient zu implementieren:

```
public void printPrimeNumbers()
```

- Fügen sie der Klasse eine Methode mit der folgenden Signatur hinzu, welche die Anzahl der Primzahlen kleiner als das "maximum" zählt. Verwenden sie ggf. Hilfsmethode wieder.

```
public int countPrimeNumbers()
```

- Erzeugen sie 8-10 Ergebnisse für verschiedene Werte von "maximum" und protokollieren sie die Ergebnisse in einem Spreadsheet (z.B. Excel oder Google Spreadsheet, wieder später der Abgabe beifügen) und lassen sie sich davon eine Kurvendarstellung generieren. Vergleichen sie das Ergebnis mit der Kurve, welche durch die Berechnung von $\frac{maximum}{\ln(maximum)}$ erzeugt wird (ln ist hier der natürliche Logarithmus).
- Fügen sie die Grafik in ihr Textdokument unter Aufgabe 5 ein.
- Vergleichen sie den Aufwand zur Berechnung der Formel mit dem Zählen durch ihren Algorithmus, schreiben sie das Ergebnis in ihr Textdokument.
- Versuchen sie die KI eine Lösung erstellen zu lassen. Ist sie anders? Besser oder schlechter? Schreiben sie das Ergebnis in ihr Textdokument.

Aufgabe 6: Erste Schritte zum Sortieren

- Erstellen sie eine neue Klasse "Sorter". Fügen sie eine Methode mit der Signatur "public void sort(int[] array)" hinzu.
- Implementieren sie die Methode so, dass nach der Ausführung array in aufsteigender Reihenfolge sortiert ist. Falls er ihnen aus der Vorlesung bereits bekannt ist, dürfen sie auch den Algorithmus "Sortieren durch Einfügen" verwenden, aber versuchen sie einmal, sich selbst eine Lösung zu überlegen.
- Testen Sie Ihre Implementation, indem sie die folgende main-Methode in der Klasse Sorter anlegen und das Programm ausführen.

```
public static final void main(String[] args)
{
    int[] array = new int[100];
    Util util = new Util();
    util.fillArrayRandom(array, 100);
    Sorter mySorter = new Sorter();
    mySorter.sort(array);
    util.printArray(array);
}
```

- Lassen sie sich von einer KI ihrer Wahl die Aufgabe lösen. Geben sie der KI dabei genug Informationen, um die Aufgabe bearbeiten zu können. Vergleichen sie ihre Lösung mit der KI Lösung. Wo liegen die Unterschiede? Was ist gleich? Schreiben sie die Antwort in ihr Textdokument unter Aufgabe 6 auf.

Aufgabe 7: Abgabe

- Erstellen sie eine .pdf-Datei aus ihrem Textdokument und laden sie es in Moodle als Gruppenabgabe hoch (nur 1x pro Kleingruppe).