Aufgabe 1

1. Das Programm nimmt ein Datum entgegen und gibt dann den dazu gehörenden Wochentag zurück.

Beispiel: „2024-01-26“ sollte „Friday“ zurückgeben

„2024-01-25“ sollte „Thursday“ zurückgeben

1. Eingabe: „2024.01.26“ wird nicht Freitag zurückgeben, da der Punkt in „input.split(„.“)“ eine spezielle Bedeutung hat und dafür sorgt, dass jedes beliebige Zeichen (bis auf den Zeilenumbruch) dafür sorgt, dass der input geteilt wird, wodurch dass Datum nicht erkannt werden kann
2. Bezeichner: einige Bezeichner (wd und fs) sind schlecht gewählt, da sie Abkürzungen sind, und somit nicht direkt verstanden werden.

JavaDoc/Kommentare: JavaDoc fehlt komplett wodurch es schwieriger wird die Funktion des Programms zu verstehen und z.B. über Vor- und Nachbedingungen informiert zu sein.

Erwartbarkeit: Der code folgt nicht der Regel der Erwartbarkeit, sondern tut viele unnötige unerwartete und nonkonforme Dinge, die für Verwirrung sorgen können und den code unverständlicher machen. Zum Beispiel dass in Zeile 27 nicht die Funktion setCalendarDate verwendet wird wie davor oder dass in Zeile 22 input.indefOf… anstatt input.contains() verwendet wird.

Aufgabe 2

Zeile 6, lexikalischer Fehler: Nach der Deklaration des Typs (hier String) folgt der Bezeichner, welcher bestimmten Regeln folgt. So muss der Bezeichner mit einem Buchstaben, Unterstrich oder Dollarzeichen beginnen und Leerzeichen sind ebenfalls nicht erlaubt. Der Fehler wird am schnellsten behoben, indem man das „?“ löscht.

Zeile 12, syntaktischer Fehler: Der Konstruktor besteht, neben der Sichtbarkeit, ausschließlich aus dem Klassennamen. „constructor“ verletzt die Syntax und führt zu einem Syntaktischen Fehler. Der Fehler lässt sich beheben, in dem man „constructor“ löscht.

Zeile 13, Stilfehler: s.equals gibt einen boolschen Wert zurück wodurch „== True“ unnötig und Stilistisch unschön ist. Den Fehler kann man einfach beheben, indem man „== True“ löscht.

Zeile 34, lexikalischer Fehler: Bezeichner beginnen mit einem Buchstaben, Unterstrich oder Dollarzeichen „100KiloPandasAndHeavierAmongAllPandasOfTheZoo“ ist also ungültig. Der Fehler wird am schnellsten behoben, indem man den Bezeichner ändert. // falls das doppelt zählt, dann stilfehler da bezeichner irreführend ist da nur pandas über 100 kilo und nicht 100 und über 100

Zeile 34, syntaktischer Fehler: Das Keyword default wird nur innerhalb von Interfaces verwendet, um eine default Methode zu deklarieren. Innerhalb von Klassen ist es syntaktisch falsch. Die „default“ Sichtbarkeit wird dargestellt, indem man kein Keyword vor der Methode verwendet. Eine Verbesserung wäre es „default“ zu „public“ zu ändern, damit man die Operation korrekt aufrufen kann.

Zeile 38, statisch semantischer Fehler: „pandasHeavierThan100Kilos. add(element);“ ist syntaktisch korrekt, allerdings semantisch falsch, da zwischen „.“ Und „add“ kein Leerzeichen kommt. Da der Fehler zur compile Zeit erkannt wird, ist er statisch. Der Fehler wird behoben, indem das Leerzeichen entfernt wird.

Zeile 49, DYNAMISCH semantischer Fehler: Die Operation feed verlangt ein Argument vom Typ Bamboon. Da in Zeile 49 kein Argument übergeben wird, kommt es zu einem semantischen Fehler, der zur Laufzeit auffällt, wodurch es ein dynamisch semantischer Fehler ist. Der Fehler lässt sich beheben, indem man ein initialisiertes Objekt vom Typ Bamboon als Argument übergibt.

Zeile 58, dynamisch semantischer Fehler: Der Rückgabetyp von getWeight() ist double, wodurch das Gewicht als Kommazahl zurückgegeben wird. Der Datentyp von sum ist int, wodurch es bei der Verrechnung des Gewichts zu einem Fehler kommt, da die Datentypen nicht kompatibel sind. Das Problem lässt sich beheben, indem man den Datentyp von sum zu double ändert.

Zeile 60, syntaktischer Fehler: Die for-schleife ist inkorrekt implementiert. Es fehlt die Abbruchbedingung (i ist kleiner als die Länge der pandas Liste) sowie die geschweiften Klammern „um“ „sum += pandas.get(i).getWeight();“

Zeile 59, Stilfehler: sum wird nur initialisiert, wenn die Liste pandas nicht leer ist. Wenn sie leer ist wird sum zwar standardmäßig mit 0 initialisiert, ist allerdings stilistisch unschön. Beheben lässt sich der Fehler, indem man sum zu Beginn direkt mit 0 initialisiert.

Zeile 7 maybe final weg aber idk

Zeile 6 maybe final hin aber idk

Zeile 12 maybe final hin aber idk

Zeile 47 for each klammer falsch? // STYLE