FRIEDRICH-SCHILLER-UNIVERSITÄT JENA

JENAER SCHRIFTEN

ZUR

MATHEMATIK UND INFORMATIK

Eingang: 03.06.2008 Math/Inf/04/08 Als Manuskript gedruckt

Arbeitsmaterial

zur Untersuchung
„Kompetenzorientierter Informatikunterricht
mit der visuellen Programmiersprache Puck“

Lutz Kohl
Abteilung Didaktik
Fakultät für Mathematik und Informatik
Friedrich-Schiller-Universität Jena
D – 07743 Jena

Lutz.Kohl@uni-jena.de

Inhalt:

Vorwort

Für das Forschungsvorhaben „Kompetenzorientierter Informatikunterricht mit der visuellen Programmiersprache Puck" wurde Arbeitsmaterial entwickelt, das den beteiligten Lehrerinnen und Lehrern zur Verfügung gestellt wurde. Der Untersuchung liegt ein dreistufiges Kompetenzmodell zum Thema Algorithmen zugrunde. Das Kompetenzmodell ist programmiersprachenunabhängig.

Die Lehrerinnen und Lehrer legten selbst fest, welche Kompetenzstufen ihre Schülerinnen und Schüler in ihrem Unterricht erreichen sollten. Sie entschieden selbst, welche Aufgaben sie im Unterricht nutzten. Zum Ende der Untersuchung wurde den Lehrerinnen und Lehrern ein Kompetenztest zugesandt, der im Unterricht eingesetzt werden sollte und von der Lehrkraft auszuwerten war. Die bei der Auswertung gewonnen Ergebnisse sollten den Lehrkräften helfen, die Schülerleistungen einzuordnen. Nach der Auswertung des Kompetenztests wurden die Lehrerinnen und Lehrer gebeten, einen Fragebogen zu ihren Erfahrungen auszufüllen. Eine Auswertung dazu erfolgt an anderer Stelle. Mit dieser Veröffentlichung werden die bereitgestellten Materialien interessierten Lehrkräften zur Verfügung gestellt und eine Diskussion in der Fachdidaktik Informatik wird ermöglicht.

Auf der nächsten Seite wird das Kompetenzmodell präsentiert, das der Untersuchung zugrunde liegt. Anschließend folgen Erläuterungen zur Untersuchung sowie Beispielaufgaben zu jeder Stufe des Kompetenzmodells. Im zweiten Teil dieses Materials wird eine Aufgabensammlung mit 70 Aufgaben, sortiert nach den drei Kompetenzstufen vorgestellt.[1] Abschließend wird ein Kompetenztest zu jeder Stufe des Kompetenzmodells präsentiert. Alle Programmieraufgaben in diesem Material können mit dem Puck-System gelöst werden. Informationen zur visuellen Programmiersprache Puck sind auf der Internetseite www.ipuck.de zusammengetragen. Hinweise und Erfahrungen zu den bereitgestellten Materialien sind willkommen.

[1] Die Aufgaben und zugehörige Musterlösungen wurden zu großen Teilen vom Autor selbst bzw. in vom Autor betreuten studentischen Arbeiten zur visuellen Programmiersprache Puck erstellt. Florian Rohde entwickelte 19 Aufgaben für das Programmiersystem Puck unter Verwendung des Zufallbausteins. Mathias Gunkel entwickelte 30 Aufgaben zu verschiedenen Themen. Die Aufgaben zum Thema Musik entwickelte zum größten Teil Friedrich Rau. Aufgabe 31 der Aufgabensammlung orientiert sich an einer Aufgabe aus der Thüringer Informatik-Abiturprüfung. Mike Erler modifizierte einige Aufgaben und ordnete sie in die Komponenten und Stufen des Kompetenzmodells ein. Außerdem entwickelte er weitere Aufgaben, insbesondere zu den Komponenten A und B des Kompetenzmodells.

Kompetenzmodell Algorithmen

Stufen / Komponenten	Stufe I *Die Schülerinnen und Schüler haben grundlegende Kompetenzen zu Algorithmen.* *Die Schülerinnen und Schüler …*	Stufe II *Die Schülerinnen und Schüler haben vertiefte Kompetenzen zu Algorithmen.* *Die Schülerinnen und Schüler …*	Stufe III *Die Schülerinnen und Schüler haben umfassendere Kompetenzen zu Algorithmen.* *Die Schülerinnen und Schüler …*
A Eigenschaften von Algorithmen	▷erklären den Algorithmusbegriff und die wesentlichen Eigenschaften von Algorithmen ▷überprüfen die wesentlichen Eigenschaften von Algorithmen in einfachen Fällen ▷nennen Probleme, die mithilfe von Algorithmen lösbar bzw. nicht lösbar sind	▷erklären den Algorithmusbegriff und die wesentlichen Eigenschaften von Algorithmen an bekannten Beispielen ▷begründen anhand dieser Eigenschaften, ob gegebene Handlungsabläufe Algorithmen sind ▷nennen Probleme, die mithilfe von Algorithmen lösbar bzw. nicht lösbar sind	▷erklären den Algorithmusbegriff und die wesentlichen Eigenschaften von Algorithmen an selbst konstruierten Beispielen ▷begründen anhand dieser Eigenschaften, ob gegebene Handlungsabläufe Algorithmen sind ▷nennen Probleme, die mithilfe von Algorithmen lösbar bzw. nicht lösbar sind
B Algorithmische Grundbausteine und Datentypen	▷erklären die algorithmischen Grundbausteine wie Variablen, Wertzuweisungen, Verzweigungen und Wiederholungen und wenden diese Erklärungen an ▷stellen die algorithmischen Grundbausteine als Pseudocode dar ▷verwenden einen numerischen Datentyp	▷erklären die algorithmischen Grundbausteine wie Variablen, Wertzuweisungen, Verzweigungen und Wiederholungen und wenden diese Erklärungen an ▷stellen die algorithmischen Grundbausteine in verschiedenen Darstellungsformen dar ▷verwenden verschiedene Datentypen	▷erklären die algorithmischen Grundbausteine wie Variablen, Wertzuweisungen, Verzweigungen, Wiederholungen und Unterprogramme mit Parametern und wenden diese Erklärungen an ▷stellen die algorithmischen Grundbausteine in verschiedenen Darstellungsformen dar und wechseln zwischen Darstellungsformen ▷verwenden verschiedene Datentypen
C Arbeit mit Algorithmen	▷lesen in Pseudocode gegebene einfache Algorithmen ▷prüfen schrittweise einfache Algorithmen mit gegebenen Beispielen ▷setzen gegebene einfache Algorithmen in Programme um ▷modifizieren und ergänzen einfache Algorithmen bzw. Programme nach Vorgaben	▷analysieren die Funktionsweise und den Leistungsumfang gegebener Algorithmen ▷prüfen Algorithmen mit gegebenen Beispielen mithilfe von Durchlauftabellen (Schreibtischtest) ▷setzen gegebene Algorithmen in Programme um ▷modifizieren und ergänzen Algorithmen bzw. Programme nach Vorgaben	▷analysieren die Funktionsweise und den Leistungsumfang gegebener komplexer Algorithmen ▷prüfen Algorithmen mithilfe von Durchlauftabellen (Schreibtischtest) und wählen dazu typische und untypische Beispiele selbst aus ▷setzen gegebene komplexe Algorithmen in Programme um ▷modifizieren und ergänzen komplexe Algorithmen bzw. Programme nach Vorgaben und nach selbst gesetzten Zielen ▷korrigieren gegebene fehlerhafte Algorithmen bzw. Programme
D Programmentwicklung	▷entwerfen einfache Programme skizzenhaft ▷implementieren einfache Programme mit einem Programmiersystem ▷testen einfache Programme anhand gegebener Eingaben auf ihre Grundfunktionalität	▷fertigen einen schriftlichen Entwurf für Programme an ▷implementieren Programme mit einem Programmiersystem benutzungsfreundlich ▷testen Programme anhand gegebener Eingaben auf ihre Funktionalität ▷reflektieren über den Lösungsweg	▷fertigen einen schriftlichen Entwurf für komplexe Programme an ▷implementieren komplexe Programme mit einem Programmiersystem benutzungsfreundlich ▷testen komplexe Programme anhand selbst gewählter Eingaben auf ihre Funktionalität ▷reflektieren über den Lösungsweg sowie über Vor- und Nachteile der Lösung ▷verbessern Programme eigenständig

Bei der Erarbeitung der Aufgaben wurden folgende Prinzipien beachtet:

- ▷ Ein Algorithmus, der nur eine Verzweigung oder eine Wiederholung enthält, wird im Allgemeinen der Stufe I zugeordnet.
- ▷ Ein Algorithmus, der mehrere, auch ineinander verschachtelte Verzweigungen und Wiederholungen enthält, wird im Allgemeinen der Stufe II oder III zugeordnet.
- ▷ Ein Algorithmus, der ein oder mehrere Unterprogramme mit Parametern enthält, wird im Allgemeinen der Stufe III zugeordnet.

DAS KOMPETENZMODELL

- Das Kompetenzmodell orientiert sich an den **Bildungsstandards** für das Fach Informatik der Gesellschaft für Informatik e.V. (GI) (Entwurf).
- Das Kompetenzmodell beschreibt die Kompetenzen, die am Ende der Klassenstufe 10 erreicht sein sollen. Der Unterricht kann in den Klassenstufen 8-10 stattfinden.
- In Stufe II und III des Kompetenzmodells sind die Kompetenzen, die bereits in Stufe I bzw. II gefordert wurden, grau dargestellt.
- Mit dem Kompetenzmodell wird Lehrkräften eine konkrete Beschreibung an die Hand gegeben, mit deren Hilfe sie entscheiden können, welche Kompetenzen die Schülerinnen und Schüler zum Inhaltsbereich „Algorithmen“ erwerben sollen. Dabei können die Lehrerinnen und Lehrer selbst festlegen, **welche** Stufe **welche** Schülerinnen und Schüler **wann** im Unterricht erreichen sollen.
- Mit dem Kompetenzmodell können die Leistungen der Schüler eingeordnet werden.

DIE BEISPIELAUFGABEN

- Zu jeder Komponente (A, B, C und D) gibt es auf jeder Stufe (I, II und III) Beispielaufgaben (also A I, A II, A III, B I, … , D III).
- Die Beispielaufgaben orientieren sich an den geforderten Kompetenzen des Kompetenzmodells.
- Die Beispielaufgaben sollen den Lehrkräften als Orientierung und den Schülerinnen und Schülern als Übung dienen. Der Kompetenztest ist ähnlich aufgebaut, wie die Beispielaufgaben.
- Nach Erfahrungen aus dem ersten Durchlauf der Untersuchung wurden die **Beispielaufgaben** kleinschrittig **in einzelne Arbeitsaufträge unterteilt**, wobei jeder Arbeitsauftrag auch eine **Punktangabe** hat. Dies soll verhindern helfen, dass Teilaufgaben überlesen, vergessen oder unvollständig beantwortet werden.
- Die Beispielaufgaben können im Unterricht eingesetzt werden.

DER KOMPETENZTEST

- Zu jeder Komponente (A, B, C und D) gibt es auf jeder Stufe (I, II und III) eine Aufgabe im Kompetenztest (also A I, A II, A III, B I, …, D III).
- Auch der Kompetenztest orientiert sich an den geforderten Kompetenzen des Kompetenzmodells.
- Der Kompetenztest wird von den Lehrkräften selbstständig durchgeführt und bewertet.
- Die Lehrerinnen und Lehrer legen dabei die Organisation des Tests **selbst** fest:
  - Sie entscheiden, **wie viel Zeit** für die Bearbeitung vorgesehen wird. Wir empfehlen 90 Minuten.
  - Sie entscheiden, ob der Test **benotet wird.**
  - Im Folgenden sind vier Möglichkeiten für die Wahl der Stufen angegeben:
    1. alle Schüler bearbeiten dieselbe Stufe,
    2. einige Schüler bearbeiten Stufe I, einige leistungsstarke Schüler bearbeiten Stufe II,
    3. die Schüler wählen selbst, welche Stufe sie bearbeiten,
    4. die Schüler wählen bei jeder Komponente (A, B, C, D) selbst, welche Stufe sie bearbeiten.

DIE AUFGABENSAMMLUNG

- Die Aufgabensammlung in diesem Material enthält zu jeder Komponente (A, B, C und D) auf jeder Stufe (I, II und III) verschiedene Aufgaben, die zur Übung im Unterricht eingesetzt werden können.
- Die Aufgaben dieser Sammlung wurden an der Universität Jena zusammengetragen und in das Kompetenzmodell eingeordnet.
- Die Lehrkräfte entscheiden, **welche** der Aufgaben **wann** im Unterricht eingesetzt werden.

DIE EIGENSCHAFTEN VON ALGORITHMEN

- Ein **Algorithmus** ist eine eindeutige, ausführbare Handlungsvorschrift endlicher Länge zur Lösung eines allgemeinen Problems.
- Das entwickelte Kompetenzmodell für die Sekundarstufe I bezieht sich auf die folgenden vier Eigenschaften, die jeder Algorithmus erfüllt:
  - **Eindeutigkeit:** Ein Algorithmus ist eindeutig, wenn an jeder Stelle des Algorithmus genau festgelegt ist, was zu tun ist.
  - **Ausführbarkeit:** Ein Algorithmus ist ausführbar, wenn jede einzelne Anweisung von Computer oder Mensch durchgeführt werden kann.
  - **Allgemeinheit:** Ein Algorithmus ist allgemeingültig, wenn er verschiedene Probleme derselben Art löst.
  - **Endlichkeit:** Ein Algorithmus ist endlich, wenn er aus einer begrenzten Anzahl von Anweisungen begrenzter Länge besteht.

DIE DATENTYPEN

- Im Kompetenzmodell wird in B II und B III gefordert, dass verschiedene Datentypen erklärt und angewendet werden können. Das bedeutet, dass den Schülerinnen und Schülern wenigstens zwei Datentypen mit den zugehörigen Werten, Relationen und Operationen bekannt sein müssen. In dieser Untersuchung sind dies Integer und Boolean.

DIE PARAMETERARTEN

- Im Kompetenzmodell wird in B III gefordert, dass Parameter erklärt und angewendet werden können. Hierzu gehört auch die Unterscheidung zwischen Wert- und Referenzparametern:

  - **Wertparameter:** Wertparametern kann ein Ausdruck des entsprechenden Datentyps als Startwert übergeben werden. Wertparameter verhalten sich innerhalb einer Prozedur wie lokale Variablen. Sie haben einen bestimmten Datentyp und einen Wert, der innerhalb der Prozedur verändert werden kann. Der Wert eines Wertparameters wird nach der Abarbeitung der Prozedur nicht an die aufrufende Stelle zurückgegeben.

  - **Referenzparameter:** Referenzparametern wird eine Variable übergeben. Der Wert des Referenzparameters kann dann in der Prozedur verändert werden. Nach Abarbeitung der Prozedur wird der Variablen der letzte Wert des Referenzparameters zugewiesen. Somit können Ergebnisse an die aufrufende Stelle zurückgegeben werden.

DIE DARSTELLUNGSFORMEN

- In dieser Untersuchung werden Pseudocodes, Struktogramme und Programmablaufpläne als Darstellungsformen für algorithmische Grundbausteine und Algorithmen genutzt.
- Im Kompetenzmodell wird in B II und B III gefordert, dass verschiedene (mindestens zwei) Darstellungsformen beherrscht werden. In dieser Untersuchung sind die Aufgaben so konstruiert, dass eine der beiden zu beherrschenden Darstellungsformen Pseudocode ist. Bei der zweiten Darstellungsform kann zwischen Struktogrammen und Programmablaufplänen gewählt werden (vgl. Beispielaufgaben B II und B III).
- In den Aufgaben der **Komponente C** werden **gegebene Algorithmen** immer in **Pseudocode** dargestellt (vgl. Beispielaufgaben C I, C II und C III).

DER PSEUDOCODE

- Pseudocode ist eine Darstellungsform für algorithmische Grundbausteine und Algorithmen.
- Es gibt **keine eindeutige Festlegung,** wie Pseudocode formuliert sein muss; wichtig ist, dass er für den Leser **verständlich** ist.
- Mit Pseudocode ist es möglich, **leistungsschwache Schülerinnen und Schüler** bei der Implementierung eines Algorithmus zu **unterstützen**. (**Beim Puck-System kann** in der Lehrerversion der **Pseudocode** eines Programmes **automatisch generiert werden.)**
- Ein Beispiel für Pseudocode:

PROGRAMM COUNTDOWN

Lege die Variable *countdown* vom Typ Integer an.
Fordere den Benutzer mit "Wie lange soll der Countdown laufen (in Sekunden)?" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable *countdown*.
Mache Folgendes so lange, bis *(countdown <= 0)* wahr ist:

> Gib den Wert von *countdown* aus.
> Warte mit der Abarbeitung des Programms *1* Sekunde.
> Weise der Variablen *countdown* den Wert von *countdown-1* zu.

Gib den Text "Countdown abgelaufen." aus und wechsele in die nächste Zeile.

- Bei der Darstellung von Pseudocode in Beispielaufgaben, Aufgabensammlung und Kompetenztest wurde stets Folgendes beachtet:
  - Ausdrücke und Variablen werden stets kursiv geschrieben
  - Schleifenrümpfe, Entscheidungszweige und Unterprogramme werden stets eingerückt in einem Kasten dargestellt

DAS STRUKTOGRAMM

- Ein Struktogramm ist eine grafische Darstellungsform für algorithmische Grundbausteine und Algorithmen.
- Ein Beispiel:

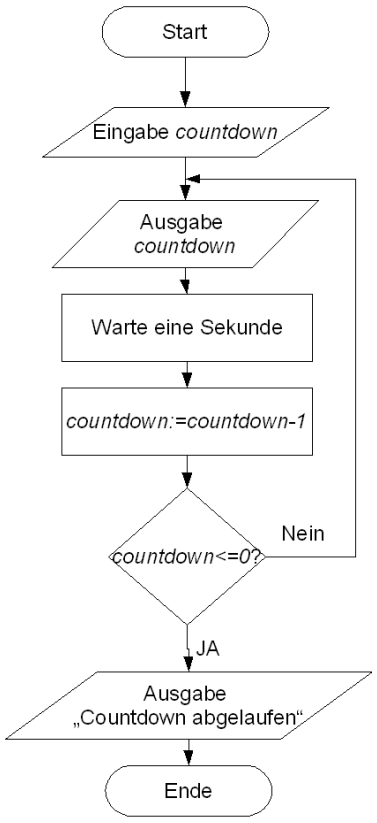
Countdown

Eingabe *countdown*	
Wieder-hole	Ausgabe *countdown*
	Warte eine Sekunde
	countdown:=countdown-1
bis *countdown<=0*	
Ausgabe „Countdown abgelaufen"	

- Weitere Informationen zu Struktogrammen gibt es unter: http://de.wikipedia.org/wiki/Nassi-Shneiderman#Sinnbilder_nach_DIN_66261

DER PROGRAMMABLAUFPLAN

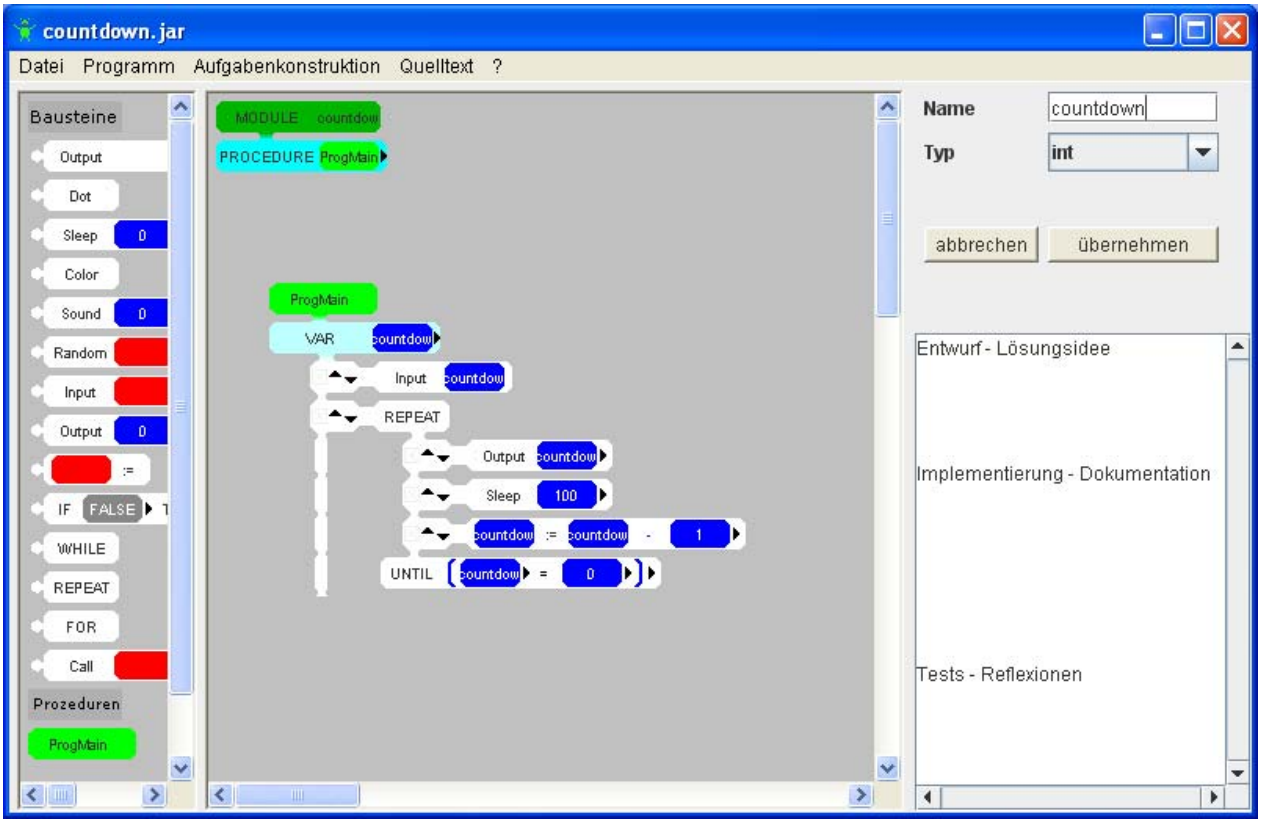
- Ein Programmablaufplan ist eine grafische Darstellungsform für algorithmische Grundbausteine und Algorithmen.
- Ein Beispiel:



- Weitere Informationen zu Programmablaufplänen gibt es unter: http://de.wikipedia.org/wiki/Programmablaufplan

DAS PUCK-SYSTEM

- Das **Puck**-System **ist für den Anfangsunterricht** in Informatik an allgemeinbildenden Schulen entwickelt worden. Es ist kein Werkzeug für die industrielle Softwareproduktion.
- **Puck-Programme sind jederzeit ausführbar, Syntaxfehler** wie vergessene Semikolons oder falsch geschriebene Variablennamen **sind nicht möglich**.
- Im Puck-System werden verschiedene Bausteine zur Verfügung gestellt (Eingabe, Ausgabe, Zuweisungen, Grafikausgaben, Tonausgaben, Zufallswerte, Kontrollstrukturen, Prozeduraufrufe).
- Die Datentypen Integer und Boolean sowie Prozeduren mit Wert- und Referenzparametern können mit dem Puck-System erlernt werden.
- Puck wurde als Open Source-Projekt entwickelt und ist somit **kostenlos** erhältlich. Das System kann also problemlos zu Hause installiert oder sogar in Projekten weiterentwickelt werden.
- Das Puck-System beinhaltet eine Dokumentation / Hilfe, in der alle Bausteine erklärt sind. Somit können sich Schülerinnen und Schüler auch selbstständig in für sie neue Bausteine einarbeiten.
- Unter www.ipuck.de sind viele Informationen zum Puck-System zusammengetragen. Schülerinnen und Schüler sowie Lehrkräfte sind eingeladen, an dieser Plattform mitzuarbeiten.
- Ein Beispielprogramm in Puck:



- Eine ausführlichere Einführung in das Puck-System mit fünf Beispielen gibt es unter: http://www.uni-jena.de/data/unijena_/faculties/minet/casio/Puck/Lutz+Kohl+Mit+Puck+einfach+Programmieren+lernen.pdf

Beispielaufgaben **Stufe I** **Komponente A**

Handlungsvorschrift: Den Flächeninhalt eines Kreises berechnen
Gegeben ist ein Kreis mit seinem Mittelpunkt.
Miss den Radius des Kreises.
Berechne das Quadrat des Radius.
Multipliziere das Quadrat des Radius mit der Zahl π (3,14).
Das Ergebnis der Multiplikation ist der Flächeninhalt des Kreises.

Erklären Sie zwei wesentliche Eigenschaften von Algorithmen und entscheiden Sie, ob die gegebene Handlungsvorschrift diese Eigenschaften erfüllt.

Eigenschaft	Erklärung	Erfüllt? Ja/Nein
		O JA O NEIN
		O JA O NEIN

4 Punkte

Beispielaufgaben **Stufe I** **Komponente B**

Erklären Sie, was man unter einer Wertzuweisung versteht.

1 Punkt

Geben Sie die Werte der Variablen a und b nach der Abarbeitung folgender Anweisungsfolge an:
Weise der Variablen a den Wert 1 zu.
Weise der Variablen b den Wert 1 zu.
Weise der Variablen a den Wert von a+1 zu.
Weise der Variablen b den Wert von a+b zu.

2 Punkte

a →
b →

Geben Sie ein Beispiel für eine Verzweigung (IF-Anweisung) als Pseudocode an.

1 Punkt

Beispielaufgaben **Stufe I** **Komponente C**

Stefans Mutter trinkt gern Tee. Verschiedene Teesorten haben unterschiedliche Ziehzeiten.
Oft vergisst Stefans Mutter den Teebeutel rechtzeitig aus ihrer Tasse zu nehmen und ärgert sich anschließend, weil ihr Tee nicht den gewünschten Geschmack hat.
Stefan hat sich den folgenden Algorithmus überlegt:

ALGORITHMUS Countdown

Lege die Variablen *minuten* und *sekunden* vom Typ Integer an.
Fordere den Benutzer mit "Wie viele Minuten?" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable *minuten*.
Weise der Variablen *sekunden* den Wert von *minuten*60* zu.

Mache Folgendes so lange, bis *(sekunden = 0)* wahr ist:

> Gib "Noch " und danach den Wert von *sekunden* aus.
> Gib den Text " Sekunden." aus und wechsele in die nächste Zeile.
> Warte mit der Abarbeitung des Programms *1* Sekunde.
> Weise der Variablen *sekunden* den Wert von *sekunden-1* zu.

Gib den Text "Der Tee ist fertig." aus.

Lesen Sie den Algorithmus und übertragen Sie ihn in ein Programm. 6 Punkte

Modifizieren Sie das Programm so, dass die Ziehzeit direkt in Sekunden eingegeben werden kann. 2 Punkte

Beispielaufgaben **Stufe I** **Komponente D**

Stefans Handyvertrag läuft aus. Er möchte einen neuen Vertrag abschließen, kann sich aber für keinen Anbieter entscheiden. Stefan telefoniert jeden Monat durchschnittlich 40 Minuten mit dem Mobiltelefon und verschickt durchschnittlich 60 SMS. Er hat sich über die Angebote verschiedener Anbieter informiert:

Anbieter	Grundgebühr in Cent	Minutenpreis in Cent	SMS-Preis in Cent
Mobilius	500	60	30
Genialion	1000	40	20
Billigus	0	80	40

Entwerfen und implementieren Sie ein Programm, das die durchschnittlichen Telefongewohnheiten eines Benutzers erfasst. Danach soll der Benutzer die Preise beliebig vieler Anbieter eingeben können. Für jeden dieser Anbieter soll das Programm die durchschnittlichen Monatskosten des Benutzers berechnen und ausgeben. 6 Punkte

Testen Sie das Programm für Stefans Telefongewohnheiten mit den gegebenen Anbietern Billigus und Genialion.
Welche durchschnittlichen Monatskosten gibt Ihr Programm aus? 2 Punkte

Billigus: Genialion:

Erklären Sie vier wesentliche Eigenschaften von Algorithmen **an Beispielen**.

4 Punkte

Handlungsvorschrift : Apfelkuchen backen
1 Becher Sahne, 3/4 Becher Zucker, 1 Tütchen Vanillezucker, 4 Eier, 2 Becher Mehl,
1 Tütchen Backpulver, Prise Salz, Äpfel
Alles vermischen (außer den Äpfeln) und auf einem Backblech verteilen.
Die Äpfel in Scheiben schneiden, drauflegen und leicht andrücken.
Im Ofen bei 200 Grad 15 bis 20 Minuten braun backen.

Ist die gegebene Handlungsvorschrift ein Algorithmus?

O JA O NEIN

1 Punkt

Begründen Sie Ihre Entscheidung.

1 Punkt

Nennen Sie je ein Beispiel für ein Problem, das mithilfe von Algorithmen lösbar bzw. nicht lösbar ist.

mithilfe von Algorithmen lösbar	
mithilfe von Algorithmen nicht lösbar	

2 Punkte

Erklären Sie den algorithmischen Grundbaustein Verzweigung (IF-Anweisung).

1 Punkt

Ein Ausschnitt aus einem Algorithmus zur Ausgabe der Preise einer Ferienwohnung ist gegeben:

```
…
Lasse den Wert der Variablen tag von 1 bis 14 in Schritten der Größe 1 laufen und mache Folgendes:
    Wenn die Bedingung (tag < 7) wahr ist, mache Folgendes:
        Weise der Variablen preis den Wert von a*60 zu.
        ...
    Wenn die Bedingung (tag < 7) falsch ist, mache Folgendes:
        Weise der Variablen preis den Wert von a*50 zu.
        ...
    …
…
```

2 Punkte

Erklären Sie, welche Aufgabe die Schleife in dem Algorithmus erfüllt.

Erklären Sie, welche Aufgabe die Verzweigung in dem Algorithmus erfüllt.

Geben Sie ein Beispiel für eine Wiederholung mit einer While-Schleife als Pseudocode und als Struktogramm oder Programmablaufplan an.

Pseudocode	Struktogramm oder Programmablaufplan

2 Punkte

Geben Sie für die folgenden Ausdrücke den jeweiligen Wert bei der Belegung a→4, b→1 an:

Ausdruck	Wert
(a<5) OR (a>6)	
(a-3) * (2+b)	
(b>1) AND (a>3)	

3 Punkte

Die Großeltern von Stefan besitzen einen kleinen Lebensmittelladen. Sie haben vor kurzem einen Computer gekauft. Sie wünschen sich nun von Stefan ein Programm, das die in die Jahre gekommene Kasse ersetzt. Dabei soll für alle eingekauften Waren eines Kunden jeweils die Menge und der Preis in Cent eingegeben werden. Am Ende soll der Gesamtpreis ausgegeben werden.
Stefan hat sich den folgenden Algorithmus überlegt:

ALGORITHMUS Kasse

Lege die Variablen *menge*, *preis* und *summe* vom Typ Integer an.
Weise der Variablen *summe* den Wert *0* zu.

Mache Folgendes so lange, bis *(menge = 0)* wahr ist:

> Fordere den Benutzer mit: "Wie viele Artikel (0 zum Beenden)?" auf, eine Zahl einzugeben. Speichere die eingegebene Zahl in die Variable *menge*.
>
> Wenn die Bedingung *(menge # 0)* wahr ist, mache Folgendes:
>
> > Fordere den Benutzer mit: "Preis in Cent:" auf, eine Zahl einzugeben.
> > Speichere die eingegebene Zahl in die Variable *preis*.
> > Weise der Variablen *summe* den Wert von *summe+menge*preis* zu.

Gib "Der Kunde muss " und danach den Wert von *summe* aus.
Gib den Text " Cent zahlen." aus.

Lesen Sie den Algorithmus und geben Sie seine Funktionsweise mit eigenen Worten wieder.

3 Punkte

Frau Meier kauft :
zwei Becher Joghurt für je 50 Cent,
vier Kiwis für je 25 Cent
und eine Mango für 99 Cent.

Überprüfen Sie den gegebenen Algorithmus für den Einkauf von Frau Meier mithilfe einer Durchlauftabelle.

menge	preis	summe

3 Punkte

Setzen Sie den Algorithmus in ein Programm um. 6 Punkte

Verändern Sie das Programm so, dass die Summe in Euro und Cent ausgegeben wird. 2 Punkte

Außerdem soll nach der Berechnung der Summe noch eingegeben werden, mit welchem Betrag der Kunde zahlt. Anschließend soll ausgegeben werden, wie viel Wechselgeld der Kassierer zurückzugeben hat. 2 Punkte

Stefan möchte sich ein neues Fahrrad für 230 Euro kaufen. Sein Opa hat angeboten, ihm das benötigte Geld zinsfrei zu leihen. Stefan möchte seinem Opa jeden Monat etwas von seinem Taschengeld zurückzahlen.
Stefan hat schon begonnen, einen Plan für die Rückzahlung zu erstellen:

Kredit: 230 Euro

Jahr	Monat	Rückzahlung	Restkredit
2007	7	20	210
2007	8	20	190
2007	9	20	170
…	…	…	…

Fertigen Sie einen schriftlichen Entwurf für ein Programm an, bei dem für einen zinslosen Kredit nach der Eingabe der Kredithöhe, des Startjahres, des Startmonats sowie der monatlichen Zahlung ein Plan für die Rückzahlung ausgegeben wird.

4 Punkte

Implementieren Sie das entworfene Programm benutzungsfreundlich. 7 Punkte

Testen Sie Ihr Programm mit den folgenden Eingaben.

A) Kredit: 200 Euro, Rückzahlung 20 Euro, Jahr 2007, Monat 11
B) Kredit: 100 Euro, Rückzahlung 30 Euro, Jahr 2008, Monat 3

Notieren Sie die Ausgaben Ihres Programms bei beiden Testfällen.

4 Punkte

Reflektieren Sie über Ihren Lösungsweg.
Welche Schwierigkeiten traten beim Entwurf und bei der Implementierung auf?
Wie wurden diese Probleme gelöst?

1 Punkt

Nennen Sie ein Beispiel für einen Algorithmus zum Thema Textverarbeitung.

1 Punkt

Erklären Sie an dem von Ihnen genannten Beispiel vier Eigenschaften von Algorithmen.

8 Punkte

Begründen Sie, warum Ihr Beispiel ein Algorithmus ist.

1 Punkt

Nennen Sie je zwei Beispiele für Probleme, die mithilfe von Algorithmen lösbar bzw. nicht lösbar sind.

mithilfe von Algorithmen lösbar	
mithilfe von Algorithmen nicht lösbar	

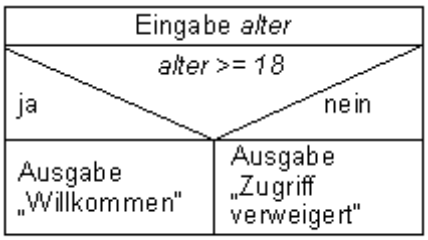
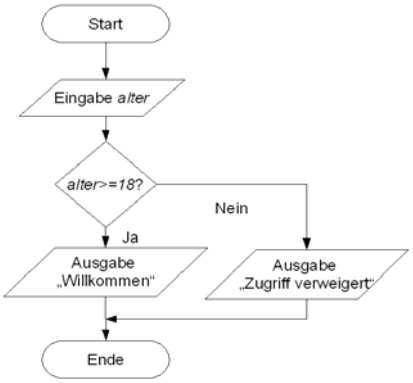
2 Punkte

Nennen Sie zwei Ihnen bekannte Parameterarten und erklären Sie die Funktionsweise der Parameterübergabe.

Parameterart	Funktionsweise der Parameterübergabe

4 Punkte

Überführen Sie die gegebenen Beispiele in **eine** andere Darstellungsform. 4 Punkte

Pseudocode	Struktogramm	Programmablaufplan
Weise der Variablen *a* den Wert von *0* zu. Solange wie *(a < 5)* wahr ist, mache Folgendes: Weise der Variablen *a* den Wert von *a+1* zu. Weise der Variablen *b* den Wert von *a*a* zu. Gib den Wert von *b* aus. Gib den Text „Programmende" aus.		
		

Geben Sie für zwei Datentypen jeweils zwei Werte und zwei Operationen an.

Datentyp	Werte	Operationen

4 Punkte

Stefan hat einen Algorithmus entwickelt:

```
ALGORITHMUS Sortieren
Prozedur sortiere2(Referenzparameter g und h vom Typ Integer)
    Lege die Variable hilf vom Typ Integer an.
    Wenn die Bedingung (g > h) wahr ist, mache Folgendes:
        Weise der Variablen hilf den Wert von h zu.
        Weise der Variablen h den Wert von g zu.
        Weise der Variablen g den Wert von h zu.

Prozedur sortiere3(Referenzparameter d,e und f vom Typ Integer)
    Rufe die Prozedur sortiere2 mit den Parametern d und e auf.
    Rufe die Prozedur sortiere2 mit den Parametern e und f auf.
    Rufe die Prozedur sortiere2 mit den Parametern d und e auf.

Hauptprogramm
    Lege die Variablen a,b und c vom Typ Integer an.
    Fordere den Benutzer mit "1. Zahl:" auf, eine Zahl einzugeben.
    Speichere die eingegebene Zahl in die Variable a.
    Fordere den Benutzer mit "2. Zahl:" auf, eine Zahl einzugeben.
    Speichere die eingegebene Zahl in die Variable b.
    Fordere den Benutzer mit "3. Zahl:" auf, eine Zahl einzugeben.
    Speichere die eingegebene Zahl in die Variable c.
    Rufe die Prozedur sortiere3 mit den Parametern a, b, c auf.
    Gib "kleinste Zahl:  " und danach den Wert von a aus.
    Gib "    mittlere Zahl: " und danach den Wert von b aus.
    Gib "    größte Zahl: " und danach den Wert von c aus.
```

Lesen Sie den Algorithmus und geben Sie seinen Leistungsumfang und seine Funktionsweise mit eigenen Worten wieder.

5 Punkte

Die Prozedur *sortiere2* enthält einen Fehler. Kreuzen Sie die fehlerhafte Zeile im Algorithmus an und schreiben Sie auf, wie diese richtig heißen müsste. 1 Punkt

Überprüfen Sie den korrigierten Algorithmus für ein typisches und ein untypisches Beispiel mithilfe von Durchlauftabellen.

6 Punkte

Setzen Sie den Algorithmus in ein Programm um. 7 Punkte

Verändern Sie das Programm so, dass auch vier Zahlen sortiert werden können. 5 Punkte

Stefans Bruder Peter spielt gerne Stein, Schere, Papier. Bei diesem Spiel wählen zwei Spieler gleichzeitig einen der drei Begriffe. Stein gewinnt gegen Schere. Schere gewinnt gegen Papier und Papier gewinnt gegen Stein. Haben beide Spieler den gleichen Gegenstand gewählt, geht das Spiel unentschieden aus.

Peter wünscht sich ein Programm, dass das Spiel „Stein, Schere, Papier“ simuliert. Dabei soll der Benutzer einen Gegenstand auswählen. Anschließend soll der Computer zufällig einen der drei Gegenstände wählen. Am Ende sollen die gewählten Gegenstände und der Ausgang des Spiels ausgegeben werden.

Fertigen Sie einen schriftlichen Entwurf für das gewünschte Programm an.

6 Punkte

Implementieren Sie das Programm benutzungsfreundlich. 8 Punkte

Testen Sie Ihr Programm mit selbst gewählten Eingaben.
Notieren Sie Eingaben und Ausgaben Ihres Tests.

2 Punkte

Diskutieren Sie Vor- und Nachteile der Computerlösung.

4 Punkte

Verbessern oder erweitern Sie das Programm in irgendeiner Weise. 4 Punkte

Aufgabensammlung

Inhaltsverzeichnis

Stufe II 37

Stufe III 55

Stufe I

Komponente A Stufe I

1. Aufgabe

Erklären Sie die Eigenschaften von Algorithmen.

allgemein:
ausführbar:
endlich:
eindeutig:

Handlungsvorschrift : Der Weg zum leckeren Kuchen
Lies das Kuchenrezept gründlich durch.
Besorge alle Zutaten und lege sie bereit.
Mische die Zutaten und folge den Anweisungen laut der Rezeptbeschreibung.
Stelle den Kuchen in den Backofen.
Hole den Kuchen aus dem Backofen.
Lasse ihn abkühlen.
Iss ein Stück Kuchen.
Schmeckt der Kuchen nicht, dann beginne mit dem ersten Schritt.

Welche Eigenschaften erfüllt die Handlungsvorschrift? Welche erfüllt Sie nicht?

erfüllt:
nicht erfüllt:

Wie wird eine genau definierte Handlungsvorschrift mit den Eigenschaften allgemein, ausführbar, endlich und eindeutig zur Lösung von Problemen genannt?

2. Aufgabe

Füllen Sie den Lückentext aus. Setzen Sie dazu die gegebenen Wörter an der richtigen Stelle ein.

Wörter:	eindeutig allgemein ausführbar endlich Algorithmus

Ein _____ist eine Verarbeitungsvorschrift, die aus einer endlichen Folge von eindeutig ausführbaren Anweisungen besteht, mit der man eine Vielzahl gleichartiger Aufgaben lösen kann.
Wenn eine Aufgabe nicht nur ein spezielles Problem, sondern eine ganze Problemklasse löst, so ist sie _____.
Des weiteren müssen die Anweisungen verständlich formuliert sein, sowie für den Befehlsempfänger (Mensch oder Maschine) _____sein.
Der Algorithmus muss _____sein, d.h. die Beschreibung der Anweisungsfolge muss in einem begrenzten Text möglich sein.
An jeder Stelle muss der Ablauf der Anweisungen _____ sein, jede Anweisung darf nur auf genau eine Art durchgeführt werden.

Handlungsvorschrift: Programmieren

Starte einen Computer.
Öffne eine Programmierumgebung.
Sprich die Aufgabe in das Mikrofon des Computers.
Warte bis das Programm von der Programmierumgebung selbstständig generiert wurde.
Gib die Lösung der Aufgabe beim Lehrer ab.

Welche Eigenschaften von Algorithmen erfüllt die Handlungsvorschrift und welche erfüllt sie nicht?

erfüllt:
nicht erfüllt:

3. Aufgabe

Ordnen Sie die folgenden Begriffe (1, 2, 3, 4) den Definitionen (a, b, c, d) zu.

1 allgemein
2 ausführbar
3 endlich
4 eindeutig

a) Die Beschreibung der Anweisungsfolge muss in einem begrenzten Text möglich sein. Der Text muss einen Anfang und ein Ende haben.
b) Die Anweisungen besitzen Gültigkeit für die Lösung einer ganzen Problemklasse, nicht nur für ein Einzelproblem. Es müssen verschiedene Situationen erfasst werden.
c) An jeder Stelle des Algorithmus ist genau festgelegt, was zu tun ist.
d) Die Anweisungen müssen verständlich formuliert sein. Die Befehlsempfänger (Mensch oder Maschine) müssen diese Anweisungen durchführen können.

1
2
3
4

Nennen Sie Beispiele bzw. Probleme, welche sich mithilfe von Algorithmen lösen bzw. nicht lösen lassen.

-mithilfe von Algorithmen lösbar:

- mithilfe von Algorithmen nicht lösbar:

Komponente B Stufe I

4. Aufgabe

Stellen Sie eine Ihnen bekannte Wiederholung / Schleife als Pseudocode dar.

Erklären Sie wozu Variablen in Algorithmen benötigt werden.

Geben Sie die Werte der Variablen *a, b* und *hilf* nach der Abarbeitung der gegeben Anweisungsfolge an:

Weise der Variablen *a* den Wert *5* zu
Weise der Variablen *b* den Wert *11* zu.

Weise der Variablen *hilf* den Wert von *a* zu.
Weise der Variablen *a* den Wert von *b* zu.
Weise der Variablen *b* den Wert von *hilf* zu.

a →
b →
hilf →

Zusatz:

Erklären Sie den Zweck dieser Anweisungsfolge. Wozu ist die Variable *hilf* notwendig?

5. Aufgabe

Gegeben ist die Anweisungsfolge:

Weise der Variablen *a* den Wert *19* zu.
Weise der Variablen *b* den Wert *9* zu.
Solange, wie (*a* # *b*) wahr ist, mache Folgendes:

Gib den Text "Noch ein Durchlauf"aus.
Weise der Variablen *a* den Wert von *a-1* zu.
Weise der Variablen *b* den Wert von *b+1* zu.

Welche algorithmischen Grundbausteine erkennen Sie in der Anweisungsfolge? Geben Sie zu jedem ein Beispiel aus der gegebenen Anweisungsfolge an.

Wie oft wird die Ausgabe „Noch ein Durchlauf“ ausgegeben?

Wie würde sich die Anzahl der Durchläufe ändern, wenn der Variablen *a* der Wert *9* und der Variablen *b* der Wert *19* zugewiesen würden?

Stellen Sie eine Wiederholung / Schleife als Pseudocode dar, mit der 10 mal die Anschrift:
Max Mustermann, Musterstraße 1, 11111 Musterhausen
ausgegeben wird.

6. Aufgabe

Gegeben ist die Anweisungsfolge:

```
1   Lege die Variable a vom Typ Integer an.
2   Lege die Variable b vom Typ Integer an.
3   Weise der Variablen a den Wert 7 zu.
4   Lasse den Wert der Variablen b von 1 bis 10 in Schritten der Größe 1 laufen und mache Folgendes:
5       Gib den Wert von b aus.
6       Gib den Text "*" aus.
7       Gib den Wert von a aus.
8       Gib den Text "=" aus.
9       Gib den Wert von a*b aus.
10      Wechsele in die nächste Zeile.
```

Schreiben Sie die algorithmischen Grundbausteine auf, die Sie erkennen. Geben Sie jeweils ein Beispiel an.

Erklären Sie, was die Wiederholung / Schleife in der 4. Zeile bewirkt.

Geben Sie eine andere Wiederholung / Schleife an einem Beispiel als Pseudocode an.

Komponente C Stufe I

7. Aufgabe (Addition)

Gegeben ist ein Algorithmus, der zwei eingegebene Zahlen addiert und das Ergebnis ausgibt.

ALGORITHMUS Addition

Lege die Variablen *a* und *b* vom Typ Integer an. Fordere den Benutzer mit "Erste Zahl:" auf, eine Zahl einzugeben. Speichere die eingegebene Zahl in die Variable *a*. Fordere den Benutzer mit "Zweite Zahl:" auf, eine Zahl einzugeben. Speichere die eingegebene Zahl in die Variable *b*. Gib "Ergebnis der Addition: " und danach den Wert von *a+b* aus.

a) Welche Ausgabe liefert der Algorithmus für die Eingaben 8 und 3?

Ausgabe:

b) Setzen Sie den Algorithmus in ein Programm um.

c) Erweitern Sie das Programm so, dass der Benutzer auch die Ergebnisse der Subtraktion und der Multiplikation ausgegeben bekommt.

8. Aufgabe (Zufallslinien)

Ein Algorithmus zum Zeichnen einer Linie mit zufällig gewähltem Start- und Endpunkt ist gegeben:

ALGORITHMUS Zufallslinien

Lege die Variablen *x1, y1, x2* und *y2* vom Typ Integer an. Weise der Variablen *x1* einen zufälligen Wert zwischen *1* und *255* zu. Weise der Variablen *y1* einen zufälligen Wert zwischen *1* und *255* zu. Weise der Variablen *x2* einen zufälligen Wert zwischen *1* und *255* zu. Weise der Variablen *y2* einen zufälligen Wert zwischen *1* und *255* zu. Zeichne die Linie, die durch die Punkte P1(*x1, x2*) und P2(*y1, y2*) gegeben ist.

a) Setzen Sie den gegebenen Algorithmus in ein Programm um.

b) Erweitern Sie Ihr Programm so, dass zusätzlich die Farbe einer Linie zufällig gewählt wird.

c) Außerdem soll das Programm mehrere Linien zeichnen.
Der Benutzer soll die Anzahl der zu zeichnenden Linien vorgeben können.

9. Aufgabe (Klassenfahrt)

Ein Algorithmus zur Berechnung der Kosten einer Klassenfahrt ist gegeben:

ALGORITHMUS Klassenfahrt

Lege die Variablen *anzahlPersonen*, *bus*, *herberge*, *anzahlUebernachtungen* vom Typ Integer an.
Fordere den Benutzer mit "Bitte geben Sie die Anzahl der Übernachtungen in der Herberge ein." auf, eine Zahl einzugeben. Speichere die eingegebene Zahl in die Variable *anzahlUebernachtungen*.
Fordere den Benutzer mit "Bitte geben Sie die Anzahl der Personen ein." auf, eine Zahl einzugeben. Speichere die eingegebene Zahl in die Variable *anzahlPersonen*.
Fordere den Benutzer mit "Bitte geben Sie die Kosten für den Bus / die Bahn (Kosten/Tag) ein." auf, eine Zahl einzugeben. Speichere die eingegebene Zahl in die Variable *bus*.
Fordere den Benutzer mit "Bitte geben Sie die Kosten für die Übernachtung (Tag/Person) ein." auf, eine Zahl einzugeben. Speichere die eingegebene Zahl in die Variable *herberge*.
Gib "Die Gesamtkosten der Klassenfahrt sind: " und danach den Wert von *((anzahlUebernachtungen+1)*bus)+(anzahlUebernachtungen*herberge*anzahlPersonen)* aus.
Gib den Text " Euro." aus.

a) Welche Ausgabe liefert der Algorithmus für eine Gruppe von 10 Personen, die 5 Übernachtungen in einer Herberge planen. Für eine Übernachtung zahlt jeder 25 €. Für die Fahrtkosten werden für jeden Tag 150 € eingeplant.

Ausgabe:

b) Setzen Sie den gegebenen Algorithmus in ein Programm um.

Erweitern Sie Ihr Programm so, dass

c) Kosten für Museumsbesuche in die Gesamtkosten einberechnet werden,

d) die Kosten ausgegeben werden, die jede Person zahlen muss.

10. Aufgabe (Farbverlauf)

Ein Algorithmus zum Zeichnen eines horizontalen Farbverlaufes ist gegeben:

ALGORITHMUS Farbverlauf

Lege die Variable *x* vom Typ Integer an.
Weise der Variablen *x* den Wert *10* zu.
Solange wie *(x <= 250)* wahr ist, mache Folgendes:
    Setze die Farbe mit den Farbwerten Rot=*0*, Grün=*0*, Blau=*x*.
    Zeichne die Linie, die durch die Punkte P1*(x, 10)* und P2*(x, 250)* gegeben ist.
    Weise der Variablen *x* den Wert von *x+1* zu.

a) Setzen Sie den gegebenen Algorithmus in ein Programm um.

Modifizieren Sie das Programm so, dass

b) ein vertikaler (von oben nach unten) Farbverlauf mit der Farbe Rot entsteht,

c) die Farben vertikal und horizontal verlaufen.

11. Aufgabe (Segelboot)

Ein Algorithmus zum Zeichnen eines fahrenden Autos ist gegeben:

ALGORITHMUS Auto

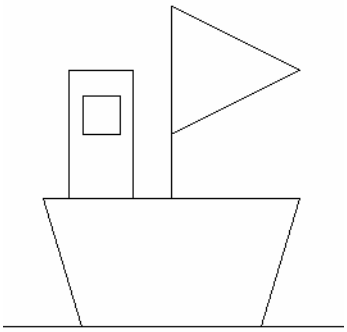
Lege die Variable *a* vom Typ Integer an.
Lasse den Wert der Variablen *a* von *0* bis *600* in Schritten der Größe *1* laufen und mache Folgendes:

> Setze die Hintergrundfarbe mit den Farbwerten Rot=*255*, Grün=*255*, Blau=*255*.
> Zeichne das durch die Punkte P1(*100+a, 100*) und P2(*350+a, 160*) gegebene Rechteck.
> Zeichne das durch die Punkte P1(*125+a, 160*) und P2(*300+a, 190*) gegebene Rechteck.
> Zeichne eine gefüllte Ellipse in das durch die Punkte P1(*120+a, 80*) und P2(*160+a, 120*) gegebene Rechteck.
> Zeichne eine gefüllte Ellipse in das durch die Punkte P1(*280+a, 80*) und P2(*320+a, 120*) gegebene Rechteck.
> Warte mit der Abarbeitung des Programms *1* hundertstel Sekunde.

a) Setzen Sie den gegebenen Algorithmus in ein Programm um.

Modifizieren Sie das Programm so, dass

b) ein Segelboot über den Bildschirm fährt,[*]



c) das Segelboot komplett aus dem Bildschirm fährt und der Benutzer die Geschwindigkeit des Segelbootes eingeben kann.

[*] *Hinweis: Machen Sie sich besonders bei grafischen Aufgaben wie dieser eine Skizze und beschriften Sie die markanten Punkte mit Koordinaten! Dies wird die sich anschließende Umsetzung deutlich erleichtern.*

12. Aufgabe (Fahne)

Ein Algorithmus zum Zeichnen eines Fahnenmastes, an dem die Deutschlandfahne nach oben gezogen wird, ist gegeben:

ALGORITHMUS Fahne

Lege die Variable *y* vom Typ Integer an.
Lasse den Wert der Variablen *y* von *100* bis *340* in Schritten der Größe *1* laufen und mache Folgendes:

Setze die Hintergrundfarbe mit den Farbwerten Rot=*255*, Grün=*255*, Blau=*255*.
Setze die Farbe mit den Farbwerten Rot=*0*, Grün=*0*, Blau=*0*.
Zeichne das durch die Punkte P1(*100, 0*) und P2(*105, 400*) gegebene gefüllte Rechteck.
Setze die Farbe mit den Farbwerten Rot=*255*, Grün=*255*, Blau=*0*.
Zeichne das durch die Punkte P1(*106, y*) und P2(*200, y+20*) gegebene gefüllte Rechteck.
Setze die Farbe mit den Farbwerten Rot=*255*, Grün=*0*, Blau=*0*.
Zeichne das durch die Punkte P1(*106, y+21*) und P2(*200, y+40*) gegebene gefüllte Rechteck.
Setze die Farbe mit den Farbwerten Rot=*0*, Grün=*0*, Blau=*0*.
Zeichne das durch die Punkte P1(*106, y+41*) und P2(*200, y+60*) gegebene gefüllte Rechteck.
Warte mit der Abarbeitung des Programms *1* hundertstel Sekunde.

a) Setzen Sie den gegebenen Algorithmus in ein Programm um.

Modifizieren Sie das Programm so, dass

b) der Nutzer des Programms wählen kann, ob die Fahne von Deutschland oder von Österreich am Fahnenmast nach oben gezogen wird.

13. Aufgabe (Gehaltserhöhung)

Die Softwarefirma IT-Triple konnte ihre Effizienz durch den Umstieg auf eine visuelle Programmiersprache steigern. Der Chef möchte deshalb seine Mitarbeiter belohnen. Die Gehälter aller Mitarbeiter sollen um 4 %, mindestens aber um 80 € im Monat erhöht werden.

Der Algorithmus zur Berechnung dieser Gehaltserhöhung ist gegeben:

ALGOTITHMUS Gehaltserhoehung

Lege die Variable *altesgehalt* vom Typ Integer an.
Fordere den Benutzer mit "Geben Sie bitte ihr altes Gehalt ein." auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable *altesgehalt*.

Wenn die Bedingung *((altesgehalt*104 DIV 100)-altesgehalt > 80*) wahr ist, mache Folgendes:
    Gib "Neues Gehalt:" und danach den Wert von (*altesgehalt*104 DIV 100*) aus.
Wenn die Bedingung (*(altesgehalt*104 DIV 100)-altesgehalt > 80*) falsch ist, mache Folgendes:
    Gib "Neues Gehalt:" und danach den Wert von *altesgehalt+80* aus.

a) Welche Ausgaben liefert der Algorithmus für das Gehalt 1000 € und für das Gehalt 10000 €?

Ausgabe:

b) Lesen Sie den Algorithmus, übertragen Sie ihn in ein Programm und testen Sie es.

c) Finden Sie durch mehrfaches Testen heraus, ab welcher Gehaltsgrenze mehr als 80 € Gehaltserhöhung fällig ist.

d) Ergänzen Sie das Programm so, dass jeweils die Gehaltserhöhung mit ausgegeben wird.

14. Aufgabe (Ameisenverkehr)

Ein Algorithmus zeichnet einen Startpunkt in der Mitte des Bildschirms. Davon ausgehend wird anschließend zufällig links, rechts, ober-, unterhalb oder schräg vom letzten Punkt ein neuer gesetzt, so dass eine Kette entsteht.
Der Algorithmus ist vorgegeben:

ALGORITHMUS Ameisenverkehr

Lege die Variablen *x*, *y*, *dx*, *dy* und *schritte* vom Typ Integer an.
Weise der Variablen *x* den Wert *400* zu.
Weise der Variablen *y* den Wert *200* zu.
Zeichne den Punkt P(*x, y*).
Fordere den Benutzer mit "Wie viele Schritte sollen ausgeführt werden?" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable *schritte*.

Mache Folgendes so lange, bis *(schritte = 0)* wahr ist:

> Weise der Variablen *dx* einen zufälligen Wert zwischen *1* und *3* zu.
> Weise der Variablen *dy* einen zufälligen Wert zwischen *1* und *3* zu.
> Weise der Variablen *x* den Wert von *x+dx-2* zu.
> Weise der Variablen *y* den Wert von *y+dy-2* zu.
> Warte mit der Abarbeitung des Programms *1* hundertstel Sekunde.
> Zeichne den Punkt P(*x, y*).
> Weise der Variablen *schritte* den Wert von *schritte-1* zu.

a) Setzen Sie den gegebenen Algorithmus, der eine verirrte Ameise simuliert, in ein Programm um.

Erweitern Sie Ihr Programm so, dass

b) der Benutzer die Startkoordinaten vorgeben kann,

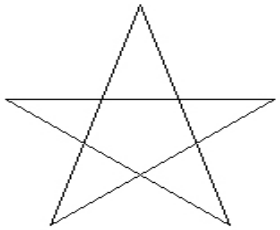
c) die Ameise sich auch sprunghaft über mehrere Felder bewegen kann,

d) sich zwei Ameisen zufällig auf dem Bildschirm bewegen.

Komponente D Stufe I

15. Aufgabe (Stern)

Entwerfen* und implementieren Sie ein Programm, das den gegebenen Stern zeichnet.
Überprüfen Sie die Funktionalität des Programms.



16. Aufgabe (Quader)

Entwerfen* und implementieren Sie ein Programm, das einen Quader in einer 3-dimensionalen Ansicht zeichnet.

Zusatz: Der Benutzer soll die Größe des Quaders in Pixeln angeben können.

17. Aufgabe (Namensausgabe)

Schreiben Sie ein Programm, das Ihren Namen buchstabenweise ausgibt. Nach der Ausgabe eines Buchstaben soll das Programm jeweils eine Sekunde warten und erst anschließend den nächsten Buchstaben ausgeben.
Überprüfen Sie die Funktionalität des Programms.

18. Aufgabe (Namenslied)

Erstellen Sie eine Tabelle mit 26 Spalten und zwei Zeilen. Füllen Sie die erste Zeile mit den Buchstaben des Alphabets und nummerieren Sie die Buchstaben in der zweiten Zeile aufsteigend.

a																							z
1																							26

Entwickeln Sie mithilfe dieser Tabelle ein Programm, das die Buchstaben Ihres Namens als Töne ausgibt, wobei die Zahlen unter den Buchstaben die jeweilige Tonhöhe angeben.
Testen Sie das Programm und variieren Sie die Instrumente.

* *Hinweis: Machen Sie sich besonders bei grafischen Aufgaben wie dieser eine Skizze und beschriften Sie die markanten Punkte mit Koordinaten! Dies wird die sich anschließende Umsetzung deutlich erleichtern.*

19. Aufgabe (Morsen)

Beim Morsen wird jeder Buchstabe mit einer Kombination aus langen und kurzen Tönen codiert. Die Tonhöhe ist dabei immer gleich. Nach jedem Buchstaben gibt es eine Pause.

Morsealphabet

A	.-	N	-.	Z	--..
AE	.-.-	O	---	SCH	----
B	-...	OE	---.	1	.----
C	-.-.	P	.--.	2	..---
D	-..	Q	--.-	3	...--
E	.	R	.-.	4	....-
F	..-.	S	...	5	.....
G	--.	T	-	6	-....
H	....	U	..-	7	--...
I	..	UE	..--	8	---..
J	.---	V	...-	9	----.
K	-.-	W	.--	0	-----
L	.-..	X	-..-		
M	--	Y	-.--		

Entwickeln Sie mithilfe des Morsealphabets ein Programm, das die Buchstaben Ihres Namens morst. Wählen Sie dazu ein geeignetes Instrument aus.

20. Aufgabe (Atbash)

Tobias möchte seiner Freundin Anika eine Botschaft zuschicken. Damit niemand die Botschaft lesen kann, will er diese verschlüsseln. Für die Verschlüsselung von Buchstaben wenden die beiden die Atbash-Kodierung an. Bei dieser Kodierung wird festgestellt wie weit jeder Buchstabe vom Beginn des Alphabets entfernt ist. Ersetzt wird der Buchstabe durch genau den Buchstaben, der die gleiche Entfernung vom Ende des Alphabetes hat. Das a wird durch das z ersetzt, das b durch das y und so weiter. „Zmrpz“ heißt also Anika.

Entwickeln Sie einen Algorithmus, der eine Zahl zwischen 1 und 100 wie bei der Atbash-Kodierung verschlüsselt. Die 1 wird durch die 100 ersetzt, die 2 durch die 99 und so weiter.

Verschlüsseln Sie die Zahlen 3, 96, 50, 37 mit diesem Algorithmus.

3	
96	
50	
37	

Setzen Sie den Algorithmus in ein Programm um und überprüfen Sie die Funktionalität des Programms. Testen Sie das Programm mit den Beispielen aus der vorherigen Teilaufgabe.

21. Aufgabe (Traumzimmer)

Ronnys Zimmer soll renoviert werden. Seine Mutter hat vorgeschlagen, dass er bei der Neugestaltung des Raumes auch seine eigenen Ideen einbringen soll. Ronny soll eine maßstabsgetreue Zeichnung anfertigen, die die neue Anordnung der Möbel zeigt. Nachdem Ronny dreimal versucht hat, eine Papierskizze zu machen, bekam er von seiner Mutter den Tipp, den Computer für die Zeichnung zu nutzen.

Schätzen Sie die Maße Ihrer Traummöbelstücke und Ihres Traumzimmers und entwickeln Sie ein Programm, das ein maßstabsgetreues Bild Ihres Traumzimmers zeigt. Die Möbel sollen zur besseren Unterscheidung in unterschiedlichen Farben dargestellt werden. Außerdem sollen der Maßstab und die Maße Ihrer neuen Möbel als Text ausgegeben werden.

22. Aufgabe (Inhaltsaustausch)

Entwickeln Sie ein Programm, bei dem der Inhalt von zwei Variablen getauscht wird.
Lassen Sie hierfür zunächst vom Nutzer zwei Zahlen abfragen, ordnen Sie diese Zahlen den Variablen a und b zu und geben Sie die Werte der Variablen am Bildschirm aus.
Vertauschen Sie anschließend den Inhalt der Variablen, und geben Sie diese erneut in derselben Reihenfolge aus.

23. Aufgabe (Pachterlös)

Ein Großbauer möchte sein Land erweitern. Dafür bietet er den Bauern im Dorf an, ihre Felder zu pachten. Da er größere Felder mit seinen modernen Geräten besser bewirtschaften kann, ist er bereit, für diese einen höheren Preis zu zahlen.
Für Felder ab 10 Hektar Größe will der Großbauer 150 € Pacht pro Hektar im Jahr bezahlen. Bei allen anderen Feldern nur 130 € pro Hektar.

Entwickeln Sie ein Programm, das zu der Eingabe einer Fläche in Hektar und der geplanten Pachtdauer in Jahren den Erlös ausgibt!

24. Aufgabe (Rechenspiel)

Entwickeln Sie ein Spiel, bei dem der Benutzer zwei zufällig gewählte natürliche Zahlen zwischen 1 und 20 erraten muss. Als Hilfe werden dem Benutzer die Ergebnisse der Addition, Subtraktion und Multiplikation der beiden Zahlen ausgegeben.

25. Aufgabe (Schulnoten)

Entwickeln Sie ein Programm das dem Benutzer nach Eingabe einer Note den Wortlaut „Sehr gut“, „Gut“, „Befriedigend“, „Ausreichend“, „Genügend“ bzw. „Ungenügend“ ausgibt.
Testen Sie das Programm mit der Eingabe jeder Schulnote.

26. Aufgabe (Zufallssätze)

Entwickeln Sie ein Programm, das zufällige Sätze ausgibt. Dabei sollen je ein Adjektiv, ein Substantiv und ein Verb zufällig ausgewählt werden.

Beispiele:
Angriffslustige Esel fauchen.
Störrische Katzen bellen.
Trompetende Mäuse fliegen.

Testen Sie das Programm mehrmals.

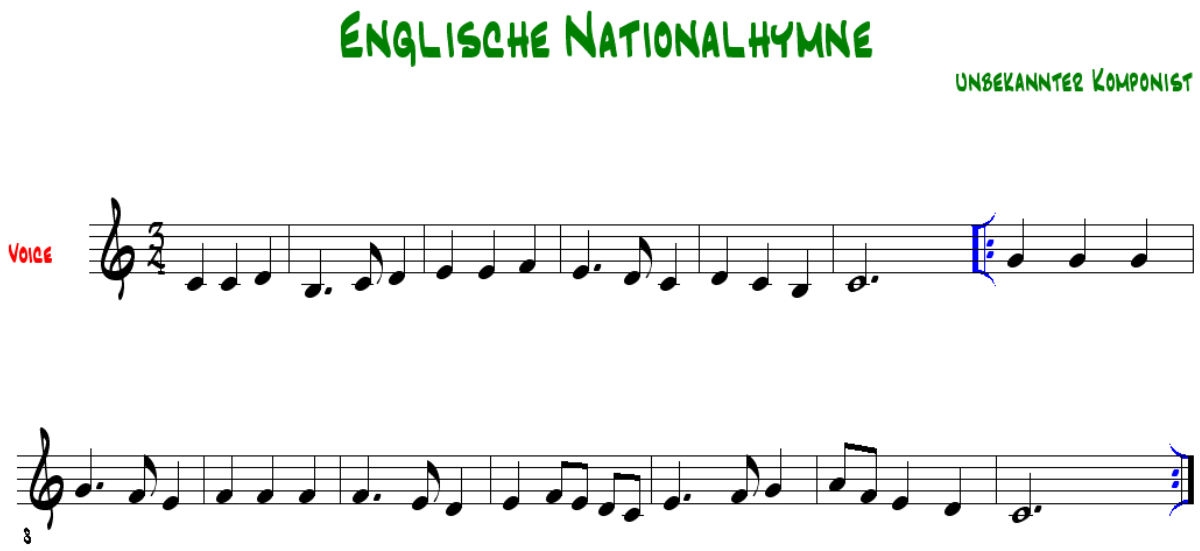
27. Aufgabe (Kartenspiel)

Eine Kartenlegerin steigt auf moderne Technik um. Sie verwendet Skat-Karten, um in die Zukunft zu schauen. Dort gibt es die „Farben“ Kreuz, Pik, Herz und Karo. Außerdem gibt es die Zahlen bzw. Figuren 7, 8, 9, 10, Bube, Dame, König, Ass. Insgesamt gibt es 32 Karten.
Entwickeln Sie ein Programm, das eine zufällig gewählte Karte wie z.B. „Herz Ass“ ausgibt.
Testen Sie das Programm mehrmals auf seine Funktionalität.

28. Aufgabe (Würfeln)

Entwickeln Sie ein Programm, das eine zufällige Zahl zwischen 1 und 6 ausgibt, also einen Würfel simuliert, und testen Sie es.
Erweitern Sie das Programm so, dass der Benutzer jeweils nach der Ausgabe des Würfelergebnisses entscheiden kann, ob noch einmal gewürfelt werden soll.

29. Aufgabe (England)



Entwerfen Sie ein Programm, das die englische Nationalhymne spielt. Beachten Sie die Wiederholungszeichen.
Setzen Sie das Programm um und realisieren Sie die Wiederholung mit einer Schleife.
Testen Sie das Programm.

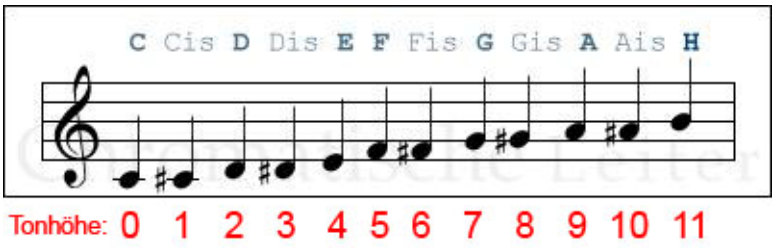
30. Aufgabe (Zufallsmusik)

Lassen Sie den Computer ein Musikstück komponieren. Die Notenanzahl, die Tonhöhen und die Tondauer sollen zufällig sein. Achten Sie darauf, geeignete Zufallsbereiche zu wählen, damit Ihr Stück nicht zu lang wird und die Tonhöhen nicht außerhalb des gültigen Bereichs liegen.

Entwerfen Sie ein Programm, das Zufallsmusik spielt.
Setzen Sie den Entwurf in ein Programm um.
Testen Sie das Programm.

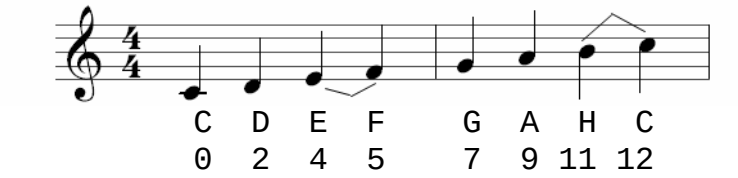
31. Aufgabe (Tonleitern)

Die chromatische Tonleiter besteht aus zwölf Tönen:

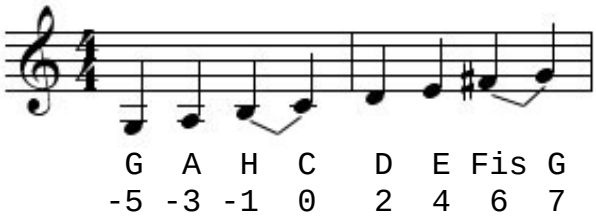


Von einem Ton dieser Tonfolge zum nächsten führt ein Halbtonschritt und zum übernächsten Ton ein Ganztonschritt.

Die C-Dur Tonleiter besteht aus folgenden Tönen:



Die G-Dur Tonleiter besteht aus den Tönen:



Eine Dur-Tonleiter besteht also aus acht Tönen. Die Tonhöhe von einem zum nächsten Ton erhöht sich jeweils um 2 (Ganztonschritt), außer vom 3. zum 4. und vom 7. zum 8. Ton, bei welchen sie sich nur um 1 (Halbtonschritt) erhöht.

Entwickeln Sie ein Programm, das vom Nutzer die Eingabe eines Grundtons (0 bis 11) als Integerwert erwartet und die entsprechende Dur-Tonleiter ausgibt.
Testen Sie das Programm indem Sie sich die D-Dur Tonleiter vorspielen lassen.

Stufe II

Komponente A Stufe II

32. Aufgabe

Erklären Sie die Eigenschaften von Algorithmen **an Beispielen**.

allgemein:
ausführbar:
endlich:
eindeutig:

Erklären Sie den Algorithmusbegriff.

Ein Algorithmus ist ....

33. Aufgabe

Handlungsvorschrift: Zähne putzen
Nimm dir deine Zahnbürste.
Öffne die Zahnpastatube.
Befeuchte deine Zahnbürste mit Wasser und trage dann Zahnpasta auf die Zahnbürste auf oder trage gleich Zahnpasta auf die Zahnbürste auf.
Putze deine Zähne in der Form, wie du es gelernt hast.
Beende das Zähneputzen nach 3 Minuten und spüle den Mund und das Waschbecken aus.

Erklären Sie die Eigenschaften von Algorithmen an Beispielen und entscheiden Sie, ob die gegebene Handlungsvorschrift diese erfüllt. Begründen Sie Ihre Entscheidung.

Eigenschaft	Erklärung	Erfüllt? Ja / Nein und Begründung

Erklären Sie den Begriff Algorithmus.

Ist die gegebene Handlungsvorschrift ein Algorithmus? Begründen Sie Ihre Antwort.

34. Aufgabe

Handlungsvorschrift 1: Einkaufen
Gehe in den Supermarkt um die Ecke.
Kaufe folgende Produkte ein: drei Flaschen Cola der Marke „Schwarzer Fuß“ (1,5 l), zwei Tafeln Schokolade der Marke “Extra Süß“ und eine Tüte Gummitiere der Marke „RIBOHA“.
Komme danach sofort wieder nach Hause und bringe mir diese Sachen.

Handlungsvorschrift 2: Einkaufen
Gehe in die Stadt und suche dir einen Laden deiner Wahl.
Kaufe mir da irgendwas Süßes zum Naschen, weil ich gerade einen Heißhunger darauf habe.
Bringe mir dann bitte die ungesunden Leckereien.

Benennen Sie die Eigenschaften von Algorithmen und erklären Sie diese anhand einer der gegebenen Handlungsvorschriften!

Welche dieser Handlungsvorschriften erfüllt die Eigenschaften von Algorithmen?

☐ Handlungsvorschrift 1 ☐ Handlungsvorschrift 2

Begründen Sie Ihre Entscheidung.

Komponente B Stufe II

35. Aufgabe

Geben Sie an, ob es sich um einen Vergleich oder um eine Wertzuweisung handelt.
Werten Sie die Vergleiche bzw. die rechte Seite der Wertzuweisungen aus.
Geben Sie weiterhin den Datentyp des Ergebnisses an.

Nutzen Sie für die Auswertung die gegebenen Belegungen der Variablen.
Weise der Variablen *a* den Wert *3* zu.
Weise der Variablen *b* den Wert *8* zu.
Weise der Variablen *x* den Wert *TRUE* zu.
Weise der Variablen *y* den Wert *FALSE* zu.

	Vergleich / Zuweisung		Ergebnis	Datentyp
d := 6 +1	☐	☐		
z := (x AND y) OR (y OR x)	☐	☐		
a = 6 +1	☐	☐		
e := 17 MOD 3	☐	☐		
a = b	☐	☐		
b = 17 DIV 3	☐	☐		
(15 MOD 3) > (2*1)	☐	☐		

Geben Sie ein selbst gewähltes Beispiel für eine Wiederholung mit einer REPEAT-UNTIL-Schleife als Pseudocode und als Struktogramm oder Programmablaufplan an.

Pseudocode	Struktogramm oder Programmablaufplan

36. Aufgabe

Erklären Sie den Unterschied zwischen (a := 13) und (a = 13).

Geben Sie ein Beispiel für eine Wiederholung / Schleife in zwei unterschiedlichen Darstellungsformen an: Pseudocode und Struktogramm oder Programmablaufplan.

Pseudocode	Struktogramm / Programmablaufplan

Erklären Sie die Funktionsweise der von Ihnen angegebenen Wiederholung / Schleife.

37. Aufgabe

Geben Sie für folgende Ausdrücke den jeweiligen Wert und den Datentyp an.

Ausdruck	Wert	Datentyp
(15 – 3 * 4) > (3 * 1)		
1 + 3 * 0		
(3 < 5) AND (5 > 3)		
(1 + 3) * 0		
(3 < 5) OR (5 > 3)		

Stellen Sie die Entscheidungen für einen Passanten an einer Fußgängerampel als Pseudocode und als Struktogramm bzw. Programmablaufplan dar.

Pseudocode	Struktogramm / Programmablaufplan

Erklären Sie den Unterschied zwischen zwei Ihnen bekannten Wiederholungen / Schleifen.

Komponente C Stufe II

38. Aufgabe (Summieren)

Ein Algorithmus zum Aufsummieren von Zahlen ist gegeben:

ALGORITHMUS Summieren

Lege die Variablen *eingabe* und *summe* vom Typ Integer an.
Weise der Variablen *summe* den Wert *0* zu.

Mache Folgendes so lange, bis *(eingabe = 0)* wahr ist:

> Fordere den Benutzer mit "Geben Sie eine Zahl ein (0 zum Abbrechen):" auf, eine Zahl einzugeben.
> Speichere die eingegebene Zahl in die Variable *eingabe*.
> Weise der Variablen *summe* den Wert von *summe+eingabe* zu.

Gib "Die Summe aller eingegebenen Zahlen ist: " und danach den Wert von *summe* aus.

a) Analysieren Sie den Algorithmus und erläutern Sie mit eigenen Worten, wie das Aufsummieren der eingegebenen Zahlen funktioniert.

b) Übertragen Sie den Algorithmus in ein Programm.

Das Programm soll zusätzlich den Durchschnitt der eingegebenen Zahlen ausgeben. (Tipp: Um den Durchschnitt zu berechnen, müssen Sie die Summe durch die Anzahl der eingegeben Zahlen teilen.)

c) Verdeutlichen Sie sich die Funktionsweise der Divisionsoperationen (DIV und MOD) für ganzzahlige Datentypen.
Welchen Durchschnitt berechnen Sie ohne Computer für die Eingaben 1, 2, 3, 4, 5, 6, 7, 8?
Kommen Sie auch auf das Ergebnis, wenn Sie die DIV-Operation nutzen?

d) Verändern Sie das Programm so, dass der Durchschnitt der eingegebenen Zahlen genauer berechnet werden kann. Nutzen Sie dazu die Operationen DIV und MOD.

Exkurs: Schriftlichen Division

```
 27:8=3,375
-24
  30
 -24
   60
  -56
    40
    40
     0
```

```
      27  DIV  8          = 3
 (3*8)=24
(27-24)=3
 (3*10)=30        (30 DIV 8) = 3 → 3,3
  (3*8)=24
 (30-24)=6
  (6*10)=60       (60 DIV 8) = 7 → 3,37
   (7*8)=56
  (60-56)=4
   (4*10)=40      (40 DIV 8) = 5 → 3,375
    (5*8)=40
   (40-40)=0
```

39. Aufgabe (Umdrehen)

Eine einfache Form Zahlen zu verschlüsseln ist, die Zahlen einfach umzudrehen. So wird 23456 zu 65432 und 547490 wird zu 094745.
Ein Algorithmus zum Umdrehen von Zahlen ist gegeben:

ALGORITHMUS Umdrehen

Lege die Variablen *zahl, u* und *v* vom Typ Integer an.
Fordere den Benutzer mit "Bitte geben Sie die Zahl zum Umdrehen ein" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable *zahl*.

Solange wie *(zahl > 0)* wahr ist, mache Folgendes:

Weise der Variablen *u* den Wert von *zahl MOD 10* zu.
Gib den Wert von *u* aus.
Weise der Variablen *v* den Wert von *zahl DIV 10* zu.
Weise der Variablen *zahl* den Wert von *v* zu.

a) Durchlaufen Sie den Algorithmus mit den Zahlen 12345 und 1978. Nutzen Sie dazu die vorgegebenen Durchlauftabellen.

zahl	*u*	*v*
12345		
Ausgabe:		

zahl	*u*	*v*
1978		
Ausgabe:		

b) Analysieren Sie den Algorithmus und beschreiben Sie mit eigenen Worten die Funktionsweise des Algorithmus.

c) Setzen Sie den gegebenen Algorithmus in ein Programm um.

d) Erweitern Sie das Programm so, dass mithilfe einer Wiederholung / Schleife mehrere aufeinanderfolgende Zahlen umgedreht und ausgegeben werden. Der Benutzer soll den Start- und Endwert des Zahlenbereiches festlegen können.

40. Aufgabe (Punktwolke)

In einem Rechteck der Größe *100 x 100* sollen Punkte mit bestimmten Wahrscheinlichkeiten gezeichnet werden. Die Wahrscheinlichkeit, dass ein Punkt gesetzt wird, soll mit steigendem x-Wert zunehmen. Ein Algorithmus ist gegeben:

ALGORITHMUS Punktwolke

```
Lege die Variable punktzeichnen vom Typ Boolean an.
Lege die Variablen x und y vom Typ Integer an.
Lasse den Wert der Variablen x von 1 bis 100 in Schritten der Größe 1 laufen und mache Folgendes:
    Lasse den Wert der Variablen y von 1 bis 100 in Schritten der Größe 1 laufen und mache Folgendes:
        Weise der Variablen punktzeichnen mit einer Wahrscheinlichkeit
        von x % den Wert TRUE zu.
        Wenn die Bedingung punktzeichnen wahr ist, mache Folgendes:
            Zeichne den Punkt P(x, y).
```

a) Setzen Sie den gegebenen Algorithmus in ein Programm um.

Modifizieren Sie das Programm so, dass sich die Punkte

b) an einer anderen Seite häufen,

c) in einer Ecke häufen,

d) sich kreisförmig um einen Punkt häufen.

Tipp: Der Abstand des Punktes (x,y) vom Zentrum (n,m) ist die Wurzel aus $(n-x)^2+(m-y)^2$. Die Wurzel muss bei der zu lösenden Aufgabe nicht gezogen werden.

***41. Aufgabe (*Zufall)**

Folgender Algorithmus ist gegeben:

ALGORITHMUS Zufall

Lege die Variablen *a, b, c, d, f* vom Typ Integer an.
Lege die Variable *e* vom Typ Boolean an.
Lasse den Wert der Variablen f von *1* bis *200* in Schritten der Größe *1* laufen und mache Folgendes:
    Weise der Variablen *a* einen zufälligen Wert zwischen 1 und 1000 zu.
    Weise der Variablen *b* einen zufälligen Wert zwischen 1 und 500 zu.
    Weise der Variablen *c* einen zufälligen Wert zwischen 1 und 1000 zu.
    Weise der Variablen *d* einen zufälligen Wert zwischen 1 und 500 zu.
    Weise der Variablen *e* mit einer Wahrscheinlichkeit von *50%* den Wert *TRUE* zu.
    Setze die Farbe mit den Farbwerten *Rot=a DIV 4*, *Grün=b DIV 2*, *Blau=c DIV 4*.
    Wenn die Bedingung *e* wahr ist, mache Folgendes:
        Zeichne das durch die Punkte P1(*a, b*) und P2(*c, d*) gegebene Rechteck.
    Wenn die Bedingung *e* falsch ist, mache Folgendes:
        Zeichne eine Ellipse in das durch die Punkte P1(*a, b*) und P2(*c, d*) gegebene Rechteck.
    Warte mit der Abarbeitung des Programms *10* hundertstel Sekunden.

a) Analysieren Sie den Algorithmus und erläutern Sie mit eigenen Worten dessen Leistungsumfang.

b) Nennen Sie eine Anwendung für diesen Algorithmus.

c) Übertragen Sie den Algorithmus in ein Programm.

42. Aufgabe (Potenzieren)

Ein Algorithmus zum Potenzieren ist gegeben:

ALGORITHMUS Potenzieren

Lege die Variablen *basis*, *ergebnis* und *exponent* vom Typ Integer an.
Fordere den Benutzer mit "Geben Sie bitte eine Zahl ein:" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable *basis*.
Fordere den Benutzer mit "Bitte geben Sie einen Exponenten ein:" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable *exponent*.
Weise der Variablen *ergebnis* den Wert 1 zu.

Solange wie *(exponent > 0)* wahr ist, mache Folgendes:

> Weise der Variablen *ergebnis* den Wert von *ergebnis*basis* zu.
> Weise der Variablen *exponent* den Wert von *exponent-1* zu.

Gib "Ergebnis des Potenzierens: " und danach den Wert von *ergebnis* aus.

a) Prüfen Sie den Algorithmus mit Hilfe der vorgegebenen Beispiele und Durchlauftabellen.

basis	exponent	ergebnis
2	6	

basis	exponent	ergebnis
6	-2	

b) Setzen Sie den Algorithmus in ein Programm um.

c) Stellen Sie sicher, dass der Algorithmus nur die Eingaben verarbeitet, die auch ein sinnvolles Ergebnis liefern können. Erweitern Sie das Programm so, dass die Eingaben vor der Berechnung überprüft werden.

.

43. Aufgabe (Gliederung)

Marcos Freundin Bianca muss eine schriftliche Hausarbeit anfertigen. Diese Arbeit soll in Abschnitte und Unterkapitel gegliedert werden. Die Nummerierung der Abschnitte und Unterkapitel empfindet Bianca als anstrengend.

Marco hat einen Algorithmus entwickelt, der die Nummerierung einer Hausarbeit automatisch vornimmt. Der Algorithmus ist gegeben:

ALGORITHMUS Gliederung

Lege die Variablen *abschnittanzahl*, *akt_abschnitt*, *unterkapitelanzahl*, *akt_unterkapitel* vom Typ Integer an.
Fordere den Benutzer mit "Wie viel Abschnitte hat deine Arbeit?" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable *abschnittanzahl*.
Weise der Variablen *akt_abschnitt* den Wert 0 zu.

Mache Folgendes so lange, bis *(akt_abschnitt = abschnittanzahl)* wahr ist:

- Weise der Variablen *akt_abschnitt* den Wert von *akt_abschnitt+1* zu.
- Fordere den Benutzer mit "Wie viel Unterkapitel hat der Abschnitt?" auf, eine Zahl einzugeben.
- Speichere die eingegebene Zahl in die Variable *unterkapitelanzahl*.
- Weise der Variablen *akt_unterkapitel* den Wert *0* zu.
- Gib den Wert von *akt_abschnitt* aus.
- Gib den Text " .Abschnitt" aus und wechseln Sie in die nächste Zeile.

Mache Folgendes so lange, bis *(akt_unterkapitel = unterkapitelanzahl)* wahr ist:

- Weise der Variablen *akt_unterkapitel* den Wert von *akt_unterkapitel+1* zu.
- Gib den Wert *von akt_abschnitt* aus.
- Gib "." und danach den Wert von *akt_unterkapitel* aus.
- Gib den Text ". Unterkapitel" aus und wechsele in die nächste Zeile.

a) Analysieren Sie den Algorithmus und erläutern Sie dessen Leistungsumfang und Funktionsweise.

b) Setzen Sie den gegebenen Algorithmus in ein Programm um.

Verändern Sie das Programm so, dass:

c) nur die Kapitelzahlen ohne die Worte „Abschnitt“ und „Unterkapitel“ ausgegeben werden,

d) eingegeben werden kann, mit welcher Zahl die Nummerierung der Abschnitte begonnen werden soll, damit auch ein 0. Kapitel möglich ist.

44. Aufgabe (Dreieckskonstruktion)

Gegeben ist folgender Algorithmus:

ALGORITHMUS Dreieckskonstruktion

Lege die Variablen *a*, *b*, *c* vom Typ Integer an.
Lege die Variable *bed1*, *bed2*, *bed3* vom Typ Boolean an.
Fordere den Benutzer dreimal mit "Geben Sie die Länge der Strecke ein! " auf, eine Zahl einzugeben.
Speichere die eingegebenen Zahlen jeweils in die Variable *a*, *b* und *c*.
Weise der Variablen *bed1* den Wert von (*a* < *b*+*c*) zu.
Weise der Variablen *bed2* den Wert von (*b* < *a*+*c*) zu.
Weise der Variablen *bed3* den Wert von (*c* < *a*+*b*) zu.

Wenn die Bedingung *bed1 UND bed2 UND bed3* wahr ist, mache Folgendes:
    Gib den Text "Aus den gegebenen Strecken kann ein Dreieck konstruiert werden." aus.
Wenn die Bedingung *bed1 UND bed2 UND bed3* falsch ist, mache Folgendes:
    Gib den Text "Es ist nicht möglich, aus den gegebenen Strecken ein Dreieck zu konstruieren." aus.

a) Analysieren Sie den Algorithmus mit Hilfe der vorgegebenen Beispiele und Durchlauftabellen.

a	*b*	*c*	*bed1*	*bed2*	*bed3*
3	9	4			
Ausgabe:					

a	*b*	*c*	*bed1*	*bed2*	*bed3*
8	6	7			
Ausgabe:					

b) Erläutern Sie mit eigenen Worten den Leistungsumfang des Algorithmus.

c) Setzen Sie den gegebenen Algorithmus in ein Programm um.

d) Erweitern Sie das Programm so, dass es zusätzlich noch ausgibt, ob die drei eingegebenen Strecken ein rechtwinkliges Dreieck bilden.
(Tipp: Erinnern Sie sich an den Satz des Pythagoras.)

45. Aufgabe (Risiko)

In dem Spiel Risiko kann ein Spieler die Wahrscheinlichkeit einstellen, mit der eine Münze Kopf zeigt. Die eingestellte Wahrscheinlichkeit ist gleichzeitig das Risiko des Spielers, das heißt es ist auch die Höhe des Geldbetrages, den er setzt.
Zeigt die Münze Kopf, so hat der Spieler verloren und er verliert den gesetzten Geldbetrag, zeigt sie Zahl, dann gewinnt der Spieler das gesetzte Geld hinzu.
Der Algorithmus für das Spiel ist gegeben:

ALGORITHMUS Risiko

Lege die Variable *risiko*, *geld* vom Typ Integer an.
Lege die Variable *kopf* vom Typ Boolean an.
Weise der Variablen *geld* den Wert *0* zu.
Fordere den Benutzer mit "Gib ein Risiko (0-100) ein" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable *risiko*.
Weise der Variablen kopf mit einer Wahrscheinlichkeit von *risiko*% den Wert *TRUE* zu.

Wenn die Bedingung *kopf* wahr ist, mache Folgendes:
- Weise der Variablen *geld* den Wert von *geld-risiko* zu.
- Gib "Schade! Dein Geld: " und danach den Wert von *geld* aus.

Wenn die Bedingung *kopf* falsch ist, mache Folgendes:
- Weise der Variablen *geld* den Wert von *geld+risiko* zu.
- Gib "Treffer! Dein Geld: " und danach den Wert von *geld* aus.

a) Analysieren Sie den Algorithmus mit Hilfe der vorgegebenen Beispiele und Durchlauftabellen.

risiko	*geld*	*kopf*
50	0	
		TRUE

risiko	*geld*	*kopf*
99	0	
		FALSE

b) Setzen Sie den gegebenen Algorithmus in ein Programm um.

Ergänzen Sie das Programm so, dass

c) der Spieler 10 Runden nacheinander spielen kann,

d) der Spieler einen Glückwunsch bzw. einen Trost, sowie die Höhe seines Gewinns bzw. Verlustes ausgegeben bekommt.

e) Testen Sie das Programm mit dem Risiko von 10 bzw. 90 in allen 10 Runden.

46. Aufgabe (Rom)

Peters Klasse ist im Geschichtsunterricht auf einen Text gestoßen, in dem römische Zahlen verwendet werden. Zur Übung hat Peters Geschichtslehrer eine Sammlung von mehreren Dezimalzahlen gegeben, die in das römische Zahlensystem überführt werden sollen. Als Beispiel hat der Geschichtslehrer die Zahl 199 gegeben, welche im römischen Zahlensystem wie folgt dargestellt wird: CLXXXXVIIII.

Römische Ziffer:	M	D	C	L	X	V	I
Dezimalzahl:	1000	500	100	50	10	5	1

Peters großer Bruder Stefan hat zur Umrechnung in das römische Zahlensystem einen Algorithmus entwickelt:

ALGORITHMUS Rom

Lege die Variable *dezimalzahl* vom Typ Integer an.
Fordere den Benutzer mit "Geben Sie eine Zahl zwischen 1 und 99 ein:" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable *dezimalzahl*.

Solange wie *(dezimalzahl >=50)* wahr ist, mache Folgendes:
- Weise der Variablen *dezimalzahl* den Wert von *dezimalzahl-50* zu.
- Gib den Text "L" aus.

Solange wie *(dezimalzahl >=10)* wahr ist, mache Folgendes:
- Weise der Variablen *dezimalzahl* den Wert von *dezimalzahl-10* zu.
- Gib den Text "X" aus.

Solange wie *(dezimalzahl >= 5)* wahr ist, mache Folgendes:
- Weise der Variablen *dezimalzahl* den Wert von *dezimalzahl-5* zu.
- Gib den Text "V" aus.

Solange wie *(dezimalzahl >= 1)* wahr ist, mache Folgendes:
- Weise der Variablen *dezimalzahl* den Wert von *dezimalzahl-1* zu.
- Gib den Text "I" aus.

a) Analysieren Sie den Algorithmus und erläutern Sie, wie die Ausgabe der Einer durch die Buchstaben „V“ und „I“ umgesetzt ist.

b) Übertragen Sie den gegebenen Algorithmus in ein Programm und testen Sie es.
Peter soll die Zahlen 54, 74, 90 und 110 ins römische Zahlensystem umrechnen. Schreiben Sie sich die Ausgaben des umgesetzten Programms auf. Sind alle Ausgaben richtig?
Nutzen sie zur Überprüfung die oben abgebildete Tabelle.

Peter ist von Stefans Algorithmus begeistert. Er hat allerdings noch zwei Änderungswünsche.
Dezimalzahlen bis 4999 sollen berechnet werden können.
Zahlen größer als 4999 sollen nicht als Eingabe akzeptiert werden.

c) Verändern Sie das von Ihnen entwickelte Programm so, dass es den Anforderungen von Peter genügt, und testen Sie es.

47. Aufgabe (Münzwurf)

Gegeben ist folgender Algorithmus:

ALGORITHMUS Muenzwurf

Lege die Variable *kopf* vom Typ Boolean an.
Lege die Variable *anz_kopf, anz_zahl, lauf* und *a* vom Typ Integer an.
Gib den Text "Kopf wird bei 1000 Würfen mit einer Wahrscheinlichkeit von 30 % geworfen." aus und wechsele in die nächste Zeile.
Lasse den Wert der Variablen lauf von *1* bis *1000* in Schritten der Größe *1* laufen und mache Folgendes:

> Weise der Variablen *kopf* mit einer Wahrscheinlichkeit von *30%* den Wert *TRUE* zu.
> Wenn die Bedingung *kopf* wahr ist, mache Folgendes:
>
> > Weise der Variablen *anz_kopf* den Wert von *anz_kopf+1* zu.
>
> Wenn die Bedingung *kopf* falsch ist, mache Folgendes:
>
> > Weise der Variablen *anz_zahl* den Wert von *anz_zahl+1* zu

Gib "Anzahl Kopf: " und danach den Wert von *anz_kopf* aus.
Wechsele in die nächste Zeile.
Gib "Anzahl Zahl: " und danach den Wert von *anz_zahl* aus.

a) Analysieren Sie den Algorithmus und erläutern Sie mit eigenen Worten dessen Leistungsumfang und Funktionsweise.

b) Welche Ausgabe erwarten Sie? Übertragen Sie den gegebenen Algorithmus in ein Programm und testen Sie das Programm mehrfach.

c) Erweitern Sie das Programm so, dass die Wahrscheinlichkeit mit der Kopf geworfen wird und die Anzahl der Würfe vom Benutzer des Programms einstellbar sind.

Komponente D Stufe II

48. Aufgabe (Kindergeburtstag)

Ein Programm zur Unterstützung von Großeltern bei Kindergeburtstagen soll entwickelt werden. Zur Eingabe des Alters und des Geschlechts eines Kindes soll das Programm einen passenden Geschenkvorschlag ausgeben.
Entwerfen Sie schriftlich einen Algorithmus für dieses Problem.
Setzen Sie das gewünschte Programm benutzungsfreundlich um, das für Kinder der Alterspannen 0-6, 7-12 und 13-18 beider Geschlechter Geschenkvorschläge macht!
Testen Sie das Programm für Mädchen und Jungen aller drei Altersstufen.

49. Aufgabe (Sortieren)

Entwerfen Sie schriftlich ein Programm, das drei Zahlen einliest und anschließend aufsteigend sortiert ausgibt.
Beispiel: Eingabe: 7, 3, 5 Ausgabe: 3, 5, 7

Setzen Sie das Programm benutzungsfreundlich um.
Testen Sie Ihr Programm mit dem gegebenen Beispiel.
Reflektieren Sie über Ihren Lösungsweg.
Welche Schwierigkeiten traten beim Entwurf und der Implementierung auf?
Wie wurden diese Probleme gelöst?

50. Aufgabe (Primzahlen)

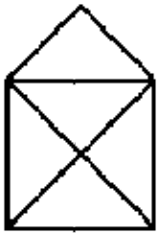
Entwerfen Sie schriftlich ein Programm, das für eine gegebene Zahl feststellt, ob es sich um eine Primzahl handelt.
Setzen Sie das Programm benutzungsfreundlich um.
Testen Sie das Programm mit den Zahlen: 2, 9, 17, 21, 23 und 25.
Reflektieren Sie über Ihren Lösungsweg.
Welche Schwierigkeiten traten beim Entwurf und der Implementierung auf?
Wie wurden diese Probleme gelöst?

51. Aufgabe (Zahlenraten)

Es soll ein Spiel entwickelt werden, bei dem eine zufällige Zahl zwischen 1 und 1000 ausgewählt wird. Anschließend darf der Benutzer zehnmal raten. Dabei soll das Programm jeweils ausgeben, ob die eingegebene Zahl größer, kleiner oder gleich der gesuchten Zahl ist.
Der Benutzer hat gewonnen, wenn er die gesuchte Zahl erraten hat. Der Ausgang des Spiels und die Anzahl der benötigten Versuche sollen ausgegeben werden.
Erarbeiten Sie einen schriftlichen Entwurf für ein Programm, das dieses Spiel realisiert.
Setzen Sie den Entwurf in ein benutzungsfreundliches Programm um.
Testen Sie das Programm.

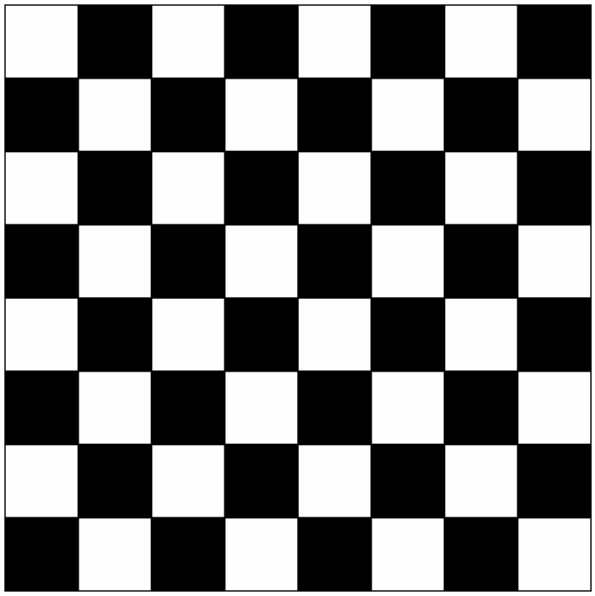
52. Aufgabe (Nikolaus)

Entwerfen Sie schriftlich ein Programm, dass das Haus vom Nikolaus Punkt für Punkt in der korrekten Reihenfolge zeichnet*.
Setzen Sie den Entwurf in ein Programm um.
Testen Sie das Programm.



53. Aufgabe (Schachbrett)

Entwerfen Sie schriftlich ein Programm, das ein Schachbrett auf dem Bildschirm zeichnet* (siehe Abbildung).



Setzen Sie den Entwurf in ein Programm um.
Testen Sie das Programm.
Reflektieren Sie über Ihren Lösungsweg.
Welche Schwierigkeiten traten beim Entwurf und der Implementierung auf?
Wie wurden diese Probleme gelöst?

* *Hinweis: Machen Sie sich besonders bei grafischen Aufgaben wie dieser eine Skizze und beschriften Sie die markanten Punkte mit Koordinaten! Dies wird die sich anschließende Umsetzung deutlich erleichtern.*

Stufe III

Komponente A Stufe III

54. Aufgabe

Geben Sie eine Handlungsvorschrift zum Thema Bildverarbeitung mit dem Computer an.

Erklären Sie anhand dieses Beispieles die Eigenschaften von Algorithmen.

Prüfen Sie, ob Ihre Handlungsvorschrift die Eigenschaften erfüllt.

55. Aufgabe

Beschreiben Sie eine Handlungsvorschrift zur Berechnung des Umfanges eines Rechteckes. Dabei sollen die einzelnen Seitenlängen vom Lehrer erfragt werden.
Achten Sie darauf, dass die Eigenschaften von Algorithmen bei der Erstellung der Handlungsvorschrift erfüllt sind.

Begründen Sie nun, warum die Eigenschaften in Ihrer Handlungsvorschrift erfüllt sind.

Nennen Sie bei zwei Eigenschaften Ihrer Wahl auch Möglichkeiten, in welchem Fall sie nicht erfüllt wären.

56. Aufgabe

Geben Sie zum Thema E-Mail zwei Handlungsvorschriften an. Dabei soll eine Handlungsvorschrift ein Algorithmus sein und die andere nicht.

Handlungsvorschrift 1:	ALGORITHMUS

Handlungsvorschrift 2:	kein ALGORITHMUS

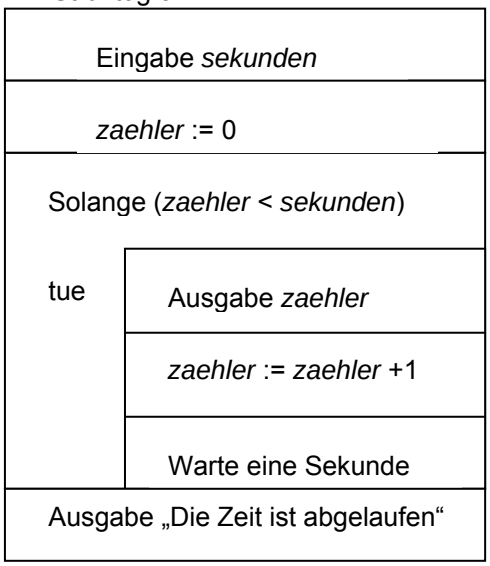
Erklären Sie die Eigenschaften von Algorithmen an den von Ihnen konstruierten Handlungsvorschriften.

Komponente B Stufe III

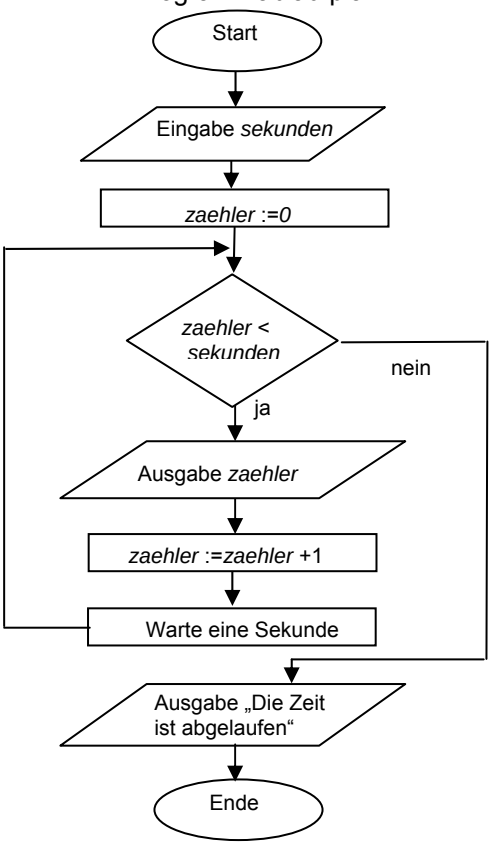
57. Aufgabe

Überführen Sie eine der gegebenen Darstellungen in Pseudocode.

Struktogramm



Programmablaufplan



Pseudocode:

In der Aufgabe wurde der Datentyp Integer verwendet.
Welchen Datentyp kennen Sie noch. Nennen Sie Operationen und Werte für diesen Datentyp.

Datentyp:	Operationen:
	Werte:

58. Aufgabe

Überführen Sie den gegebenen Pseudocode in eine andere Ihnen bekannte Darstellungsform (Struktogramm / Programmablaufplan).

Fordere den Benutzer mit "Geben Sie bitte eine Zahl ein." auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable *a*.
Weise der Variablen *b* den Wert von *a MOD 2* zu.
Wenn die Bedingung (*b = 0*) wahr ist, mache Folgendes:
  Gib den Text "gerade Zahl" aus.
Wenn die Bedingung (*b = 0*) falsch ist, mache Folgendes:
  Gib den Text "ungerade Zahl" aus.

Welche Ausgaben liefert das Programm für folgende Eingaben:

34	
65	
7	

Stellen Sie zwei Ihnen bekannte Datentypen mit Werten und Operationen gegenüber.

Datentyp	Werte	Operationen

59. Aufgabe

Geben Sie alle Ausgaben des Pogramms an.

Prozedur *Proc1*(Wertparameter *c* vom Typ Boolean, Referenzparameter *d* vom Typ Boolean)

```
Wenn die Bedingung c wahr ist, mache Folgendes:
   Weise der Variablen c den Wert FALSE zu.
   Weise der Variablen d den Wert TRUE zu.
Wenn die Bedingung c falsch ist, mache Folgendes:
   Weise der Variablen c den Wert TRUE zu.
   Weise der Variablen d den Wert FALSE zu.
```

Prozedur *Ausgabe*(Wertparameter *wert1* vom Typ Boolean, Wertparameter *wert2* vom Typ Boolean)

```
Wenn die Bedingung wert1 wahr ist, mache Folgendes:
   Gib den Text "true" aus und wechsele in die nächste Zeile.
Wenn die Bedingung wert1 falsch ist, mache Folgendes:
   Gib den Text "false" aus und wechsele in die nächste Zeile.
Wenn die Bedingung wert2 wahr ist, mache Folgendes:
   Gib den Text "true" aus und wechsele in die nächste Zeile.
Wenn die Bedingung wert2 falsch ist, mache Folgendes:
   Gib den Text "false" aus und wechsele in die nächste Zeile.
```

Prozedur *ProgMain*()

```
Lege die Variable a vom Typ Boolean an.
Lege die Variable b vom Typ Boolean an.
Weise der Variablen a den Wert TRUE zu.
Weise der Variablen b den Wert FALSE zu.
Rufe die Prozedur Proc1 mit den Parametern a, b auf.
Rufe die Prozedur Ausgabe mit den Parametern a, b auf.
Rufe die Prozedur Proc1 mit den Parametern b, a auf.
Rufe die Prozedur Ausgabe mit den Parametern a, b auf.
```

1. Ausgabe	
2. Ausgabe	
3. Ausgabe	
4. Ausgabe	

Erklären Sie die verwendeten Parameter in der Prozedur *Proc1*.

Komponente C Stufe III

60. Aufgabe (Dualzahlen)

Zum Rechnen im Computer werden meist Dualzahlen verwendet. Diese haben nicht zehn Ziffern (0, 1, 2,..., 9), sondern nur die zwei Ziffern 0 und 1. Ein Programm zum Umrechnen von Dezimalzahlen in Dualzahlen ist gegeben:

ALGORITHMUS Dualzahlen

Lege die Variable *dezi*, *potenz2*, *dual* vom Typ Integer an.
Fordere den Benutzer mit "Bitte geben Sie eine Dezimalzahl ein:" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable *dezi*.
Weise der Variablen *potenz2* den Wert *1* zu.

Solange wie *(potenz2*2 <= dezi)* wahr ist, mache Folgendes:
- Weise der Variablen *potenz2* den Wert von *potenz2**2 zu.

Gib den Text "Die entsprechende Dualzahl ist: " aus.

Solange wie *(potenz2 > 0)* wahr ist, mache Folgendes:
- Weise der Variablen *dual* den Wert von *dezi DIV potenz2* zu.
- Gib den Wert von *dual* aus.
- Weise der Variablen *dezi* den Wert von *dezi-(potenz2*dual)* zu.
- Weise der Variablen *potenz2* den Wert von *potenz2 DIV 2* zu.

a) Setzen Sie den gegebenen Algorithmus in ein Programm um und versuchen Sie, die Arbeitsweise mit Hilfe von Testausgaben zu verstehen.

b) Füllen Sie die folgenden Durchlauftabellen mit selbst gewählten Dezimalzahlen anlog der Beispieldurchlauftabelle aus. Überprüfen Sie Ihr Ergebnis anhand der Ausgabe des Programms.

dezi	potenz2	Ausgabe
3	1	
	2	
		1
	1	
		1

dezi	potenz2	Ausgabe

dezi	potenz2	Ausgabe

c) Entwerfen und implementieren Sie ein Programm, das eingegebene Dezimalzahlen in das Oktalsystem (0, 1, 2,…, 7) umrechnen kann. Orientieren Sie sich an dem Programm Dualzahlen.

d) Testen Sie das Programm Oktalzahlen mit typischen und untypischen Eingaben und überprüfen Sie die Richtigkeit der Ausgaben.

e) Entwicklen Sie eine Prozedur, in der eine Dezimalzahl in eine Zahl eines beliebigen Zahlensystems (2-9) umgewandelt und ausgegeben wird. Die Prozedur soll die Wertparameter *dezimalzahl, zahlensystem* übergeben bekommen.
Testen Sie die Prozedur ausführlich.

61. Aufgabe (Kleingeld)

Die Großeltern von Stefan besitzen einen kleinen Lebensmittelladen. Oft kommen Kinder in den Laden, um Ein- oder Zweieuromünzen in Kleingeld zu wechseln. Das Programm wechselt eine 1- oder 2-Euromünze in eine beliebige Stückelung von 50-, 20-, 10-, 5-, 2- und 1-Centmünzen. Stefans Oma legt besonderen Wert darauf, dass die Kinder genauso viel Geld herausgegeben bekommen, wie sie abgegeben haben.

Der von Stefan entwickelte Algorithmus ist gegeben:

```
ALGORITHMUS Kleingeld
  Prozedur gib_Muenzen(Wertparameter muenzwert vom Typ Integer, Referenzparameter restgeld
  vom Typ Integer)
      Lege die Variable anzahl vom Typ Integer an.

      Mache Folgendes so lange, bis (restgeld >= muenzwert*anzahl) wahr ist:
          Gib den Text "Wie viele " aus.
          Gib den Wert von muenzwert aus.
          Gib den Text " Cent Münzen wollen Sie?" aus und wechsele in die nächste
          Zeile.
          Fordere den Benutzer mit "Wie viele Münzen möchten Sie?" auf, eine Zahl
          einzugeben.
          Speichere die eingegebene Zahl in die Variable anzahl.

          Wenn die Bedingung (restgeld < muenzwert*anzahl) wahr ist, mache
          Folgendes:
              Gib den Text "Das ist zuviel!" aus und wechsele in die nächste
              Zeile.
      Weise der Variablen restgeld den Wert von restgeld-muenzwert*anzahl zu.

  Hauptprogramm()
      Lege die Variablen euro und rest vom Typ Integer an.
      Fordere den Benutzer mit "Geben Sie den Münzwert in € ein (1 oder 2)!" auf, eine Zahl
      einzugeben.
      Speichere die eingegebene Zahl in die Variable euro.
      Weise der Variablen rest den Wert von euro*100 zu.
      Rufe die Prozedur gib_Muenzen mit den aktuellen Parametern 50 und rest auf.
      Rufe die Prozedur gib_Muenzen mit den aktuellen Parametern 20 und rest auf.
      Rufe die Prozedur gib_Muenzen mit den aktuellen Parametern 10 und rest auf.
      Rufe die Prozedur gib_Muenzen mit den aktuellen Parametern 5 und rest auf.
      Rufe die Prozedur gib_Muenzen mit den aktuellen Parametern 2 und rest auf.
      Rufe die Prozedur gib_Muenzen mit den aktuellen Parametern 1 und rest auf.
```

a) Analysieren Sie den Algorithmus und erläutern Sie dessen Funktionsweise.

b) Prüfen Sie den Algorithmus anhand einer Durchlauftabelle mit selbst gewählten Testwerten.

Prozeduraufruf	euro	rest	muenzwert	restgeld	anzahl
gib_Muenzen(,)					

d) Übertragen Sie den gegebenen Algorithmus in ein Programm und testen Sie es.

Stefans kleiner Bruder Peter hat das Programm ausprobiert und hat zu wenig Geld bekommen.

e) Wie ist das möglich? Berichtigen Sie das Programm, damit die Kinder genauso viel Geld bekommen, wie sie zum Wechseln gegeben haben.

Stefans Großeltern haben noch zwei Änderungswünsche:
Vor der Wahl der Münzanzahl soll ausgegeben werden, wie viele Münzen des aktuellen Münzwertes der Benutzer maximal bekommen kann.
Bei der Eingabe der zu tauschenden Münze soll sichergestellt werden, dass für den Münzwert nur 1 oder 2 ausgewählt werden kann.

f) Verändern Sie das Programm so, dass es den Anforderungen von Stefans Großeltern genügt, und testen Sie es.

62. Aufgabe (Binäruhr)

Gegeben ist folgender Algorithmus, der eine Binäruhr realisiert.

```
ALGORITHMUS Binaeruhr

Prozedur ausg(Wertparameter zahl vom Typ Integer, Wertparameter x vom Typ Integer)
    Setze die Farbe mit den Farbwerten Rot=255, Grün=255, Blau=255.
    Zeichne das durch die Punkte P1(x, 0) und P2(x+50, 400) gegebene gefüllte Rechteck.
    Setze die Farbe mit den Farbwerten Rot=0, Grün=0, Blau=0.

    Wenn die Bedingung (zahl >= 8) wahr ist, mache Folgendes:
        Zeichne das durch die Punkte P1(x, 300) und P2(x+50, 350) gegebene gefüllte Rechteck.
        Weise der Variablen zahl den Wert von zahl-8 zu.

    Wenn die Bedingung (zahl >= 4) wahr ist, mache Folgendes:
        Zeichne das durch die Punkte P1(x, 200) und P2(x+50, 250) gegebene gefüllte Rechteck.
        Weise der Variablen zahl den Wert von zahl-4 zu.

    Wenn die Bedingung (zahl >= 2) wahr ist, mache Folgendes:
        Zeichne das durch die Punkte P1(x, 100) und P2(x+50, 150) gegebene gefüllte Rechteck.
        Weise der Variablen zahl den Wert von zahl-2 zu.

    Wenn die Bedingung (zahl >= 1) wahr ist, mache Folgendes:
        Zeichne das durch die Punkte P1(x, 100) und P2(x+50, 150) gegebene gefüllte Rechteck.
        Weise der Variablen zahl den Wert von zahl-2 zu.

    Zeichne das durch die Punkte P1(x, 0) und P2(x+50, 50) gegebene gefüllte Rechteck.

Prozedur ProgMain()
    Lege die Variable s1, s2 vom Typ Integer an.
    Lasse den Wert der Variablen s1 von 0 bis 5 in Schritten der Größe 1 laufen und mache Folgendes:
        Lasse den Wert der Variablen s2 von 0 bis 9 in Schritten der Größe 1 laufen und mache
        Folgendes:
            Warte mit der Abarbeitung des Programms 100 hundertstel Sekunden.
            Rufe die Prozedur ausg mit den Parametern s1, 200 auf.
            Rufe die Prozedur ausg mit den Parametern s2, 300 auf.
            Gib den Wert von s1 aus.
            Gib den Wert von s2 aus.
            Gib den Text " Sekunden" aus.
            Wechsele in die nächste Zeile.
```

a) Informieren Sie sich im Internet über die Funktionsweise einer Binäruhr

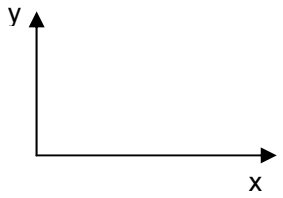
b) Analysieren Sie den Algorithmus mithilfe der angegebenen Durchlauftabelle, die die Abarbeitung der Schleifen in der Prozedur *ProgMain* näher betrachtet.

s1	*s2*	*Prozeduraufruf*
0	0	ausg(0,200); ausg (0,300)

c) Analysieren Sie an selbst gewählten Beispielen zwei unterschiedliche Aufrufe der Prozedur *ausg*. Zeichnen Sie auch die Ausgabe rechts neben die Durchlauftabellen.

zahl	*x*

zahl	*x*



d) Übertragen Sie den gegebenen Algorithmus in ein Programm und testen Sie das Programm.

e) Verändern Sie das Programm so, dass es zusätzlich zu den Sekunden auch Minuten ausgeben kann.

f) Verbessern oder erweitern Sie das Programm in irgendeiner Weise.

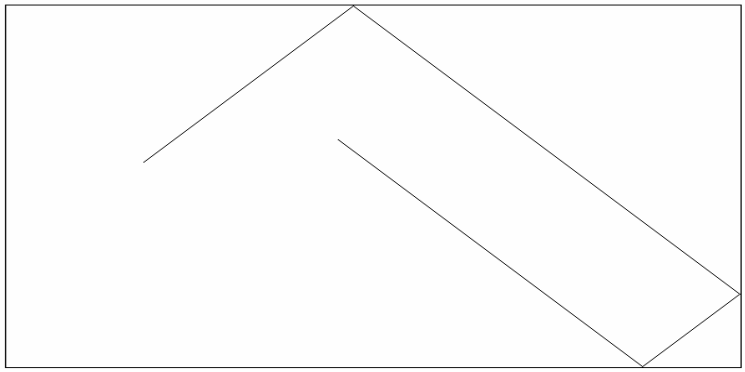
Komponente D Stufe III

63. Aufgabe (Ampel)

Entwerfen* Sie schriftlich ein Programm, dass die Schaltvorgänge einer Ampel simuliert. Für die Ampel sollen folgende Phasen realisiert werden: rot / rot gelb / grün / gelb.
Nutzen Sie dazu eine Prozedur ampelschaltung(Wertparameter rot vom Typ Boolean, Wertparameter gelb vom Typ Boolean, Wertparameter gruen vom Typ Boolean). Diese Prozedur zeichnet die jeweilige Farbe, wenn der aufrufende Parameter TRUE ist.
Setzen Sie den Entwurf in ein Programm um. Simulieren Sie mehrere Durchläufe der vier Ampelphasen.
Testen Sie das Programm.
Erweitern Sie das Programm in irgendeiner Weise.

64. Aufgabe (Billard)

Entwerfen Sie schriftlich ein Programm, bei dem der Verlauf einer Kugel auf einem Billardtisch Punkt für Punkt aufgezeichnet* wird (siehe Abbildung). Die Startkoordinaten der Kugel sollen vom Benutzer vorgegeben werden können. Die Kugel soll immer nach rechts oben gestoßen werden, sie bewegt sich stets mit einem Winkel von 45 Grad auf die Bande zu.

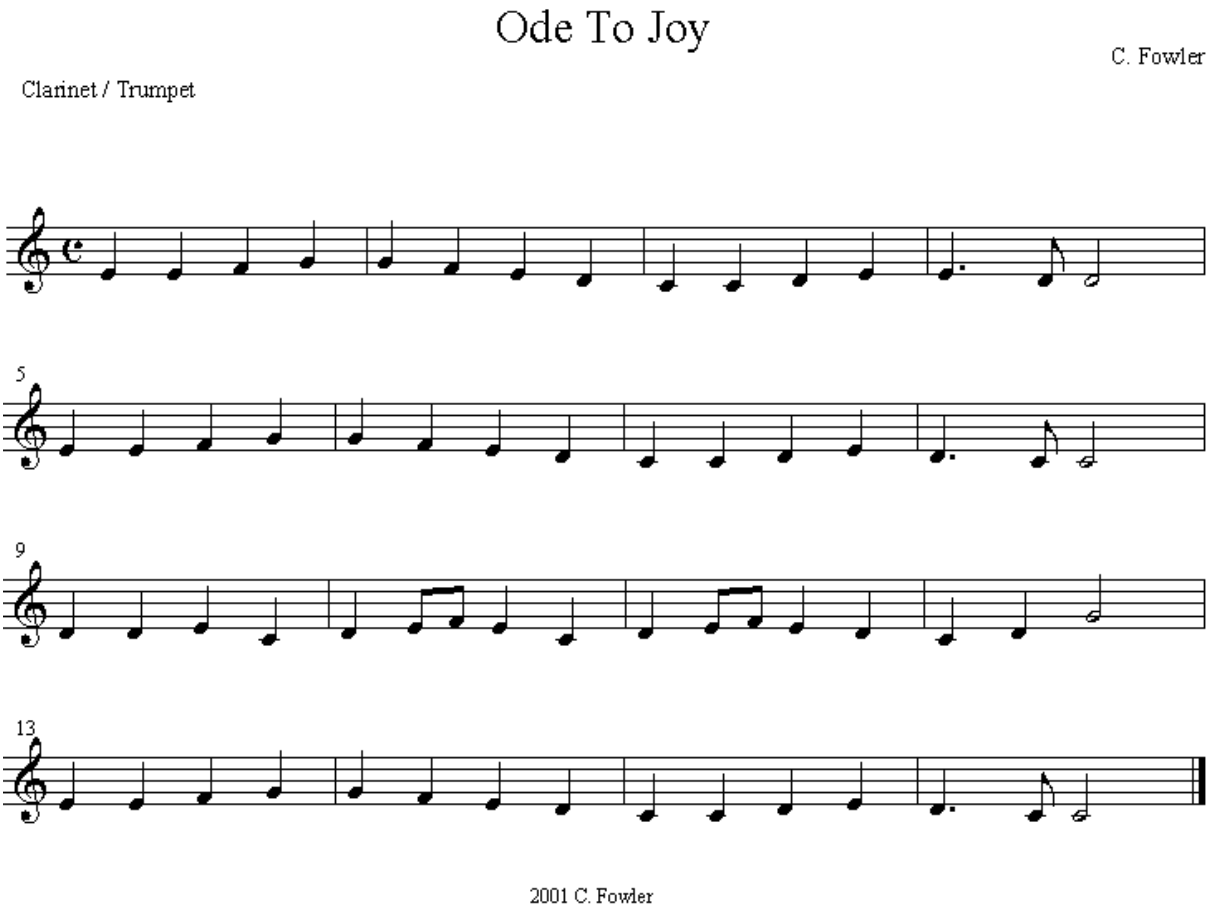


Setzen Sie den Entwurf in ein Programm um.
Testen Sie das Programm.

Verbessern oder erweitern Sie das Programm in irgendeiner Weise.

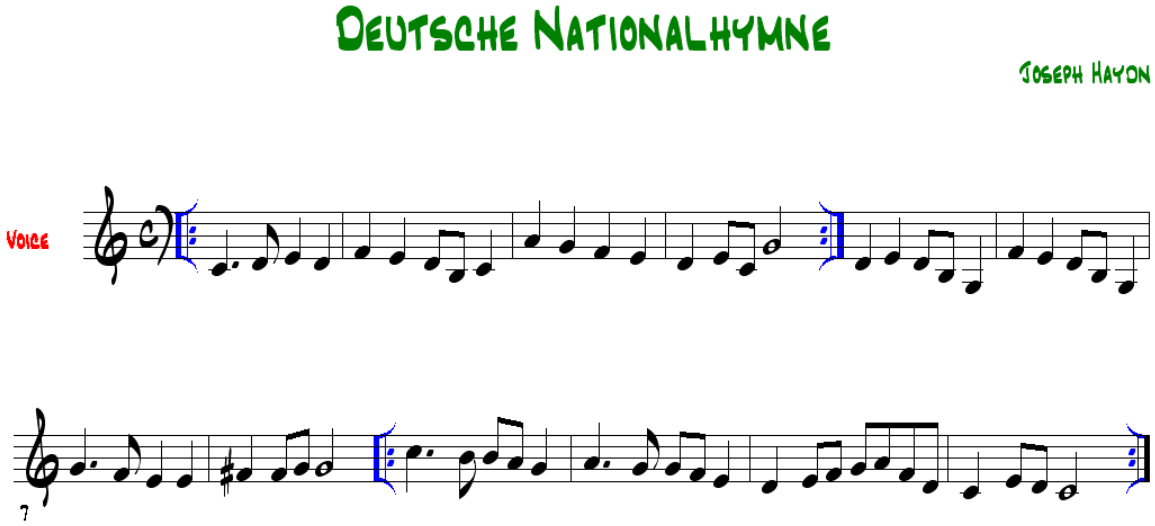
* *Hinweis: Machen Sie sich besonders bei grafischen Aufgaben wie dieser eine Skizze und beschriften Sie die markanten Punkte mit Koordinaten! Dies wird die sich anschließende Umsetzung deutlich erleichtern.*

65. Aufgabe (Götterfunken)



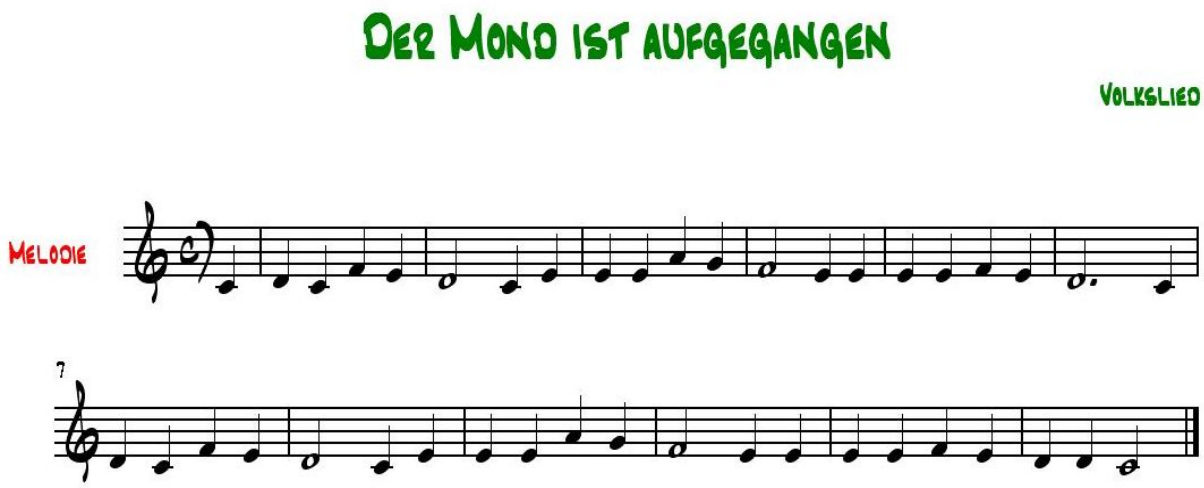
Entwerfen Sie ein Programm, das Ludwig van Beethovens: „Ode an die Freude“ spielt. Bereiche innerhalb des Liedes, die sich wiederholen, sollen über eine Prozedur realisiert werden.
Setzen Sie das Programm um.
Testen Sie das Programm.
Reflektieren Sie über Vor- und Nachteile des Einsatze von Prozeduren in Ihrer Lösung.

66. Aufgabe (Deutschland)



Entwerfen Sie ein Programm, das die deutsche Nationalhymne spielt. Beachten Sie die Wiederholungszeichen und die Erhöhung der F-Noten durch das # ab dem 8. Takt.
Bereiche innerhalb des Liedes, die sich wiederholen, sollen über eine Prozedur realisiert werden.
Setzen Sie das Programm um.
Testen Sie das Programm.

67. Aufgabe (Mondlied)



Entwerfen Sie ein Programm, das „Der Mond ist aufgegangen“ spielt.
Bereiche innerhalb des Liedes, die sich wiederholen, sollen über eine Prozedur realisiert werden.
Der Benutzer des Programms soll die Tonhöhe des tiefsten Tons und die Spieldauer der kürzesten Note festlegen können.
Überprüfen Sie mit selbst gewählten Eingaben, ob das Programm den genannten Anforderungen entspricht.

68. Aufgabe (Irrfahrt)

Eine Irrfahrt soll simuliert werden. Dabei bewegen sich die Herumirrenden von einem Startpunkt in der Mitte des Bildschirms aus immer um einen Pixel weiter. Nach jedem Schritt wird neu entschieden, ob die Richtung gewechselt werden soll. Die Wahrscheinlichkeit für einen Richtungswechsel wird am Anfang vom Benutzer festgelegt. Bei einem Richtungswechsel wird zufällig eine neue Richtung gewählt. Die Herumirrenden sollen den Bildschirm nicht verlassen.
Entwerfen Sie schriftlich ein Programm, das die gegebene Simulation einer Irrfahrt realisiert.
Setzen Sie den Entwurf in ein Programm um.
Testen Sie das Programm.
Verbessern oder erweitern Sie das Programm in irgendeiner Weise.

69. Aufgabe (Lotto)

Entwerfen Sie schriftlich ein Programm, das wie beim Lotto sechs Zahlen aus 49 ausgibt. Bei den gezogenen Zahlen dürfen natürlich keine doppelt vorkommen.
Setzen Sie den Entwurf in ein Programm um.
Testen Sie das Programm.
Reflektieren Sie über Ihren Lösungsweg, sowie über Vor- und Nachteile Ihrer Lösung.

70. Aufgabe (Brüche)

Entwerfen Sie schriftlich ein Programm, das zwei gemeine Brüche einliest und diese beiden Brüche addiert, subtrahiert, multipliziert und dividiert. Realisieren Sie die Berechnungen in einzelnen Prozeduren.
Setzen Sie den Entwurf in ein Programm um.
Testen Sie das Programm an selbst gewählten Beispielen.
Reflektieren Sie über Ihren Lösungsweg, sowie über Vor- und Nachteile Ihrer Lösung.

Tipp: Wenn für die Multiplikation und Addition von Brüchen Prozeduren angelegt werden, so kann mit diesen Prozeduren auch subtrahiert und dividiert werden, dafür müssen nur die Parameter anders übergeben werden.

Zusatz: Schreiben Sie eine Prozedur, die das Ergebnis vor der Ausgabe kürzt.

Hinweise zur Bearbeitung des Kompetenztests

Bearbeitungszeit:

- Für den Kompetenztest werden 90 Minuten Bearbeitungszeit empfohlen.

Auswahl der Aufgaben:

1. Variante: Die Lehrperson legt eine Stufe fest, die von der ganzen Klasse einheitlich bearbeitet werden soll.
2. Variante: Die Lehrperson legt individuell für jede Schülerin und jeden Schüler (oder für Schülergruppen) fest, welche Stufe bearbeitet werden soll.
3. Variante: Jede Schülerin und jeder Schüler legt für sich fest, welche Stufe bearbeitet werden soll.
4. Variante: Jede Schülerin und jeder Schüler legt für sich für jede der Komponenten die Stufe fest, die bearbeitet werden soll.

- Wenn die Schülerinnen und Schüler vor Ablauf der Zeit fertig sind, können Sie die Aufgaben einer weiteren Stufe bearbeiten.

Punktverteilung:

	Stufe I	Stufe II	Stufe III
A	4	8	12
B	4	8	12
C	8	16	24
D	8	16	24
Gesamtpunktzahl	**24**	**48**	**72**

Bewertung:

- Der Kompetenztest wird von der Lehrperson kontrolliert und bewertet.
- Es fließen bei jeder der Komponenten A, B, C und D nur die Aufgaben einer Stufe in die Bewertung ein. (Die Auswahl regelt die Lehrperson.)
- Wir schlagen vor, die Einschätzung nach folgenden Kriterien vorzunehmen:
    - Um **Stufe I** zu erreichen, müssen mindestens **18 Punkte** erreicht werden.
    - Um **Stufe II** zu erreichen, müssen mindestens **36 Punkte** erreicht werden.
    - Um **Stufe III** zu erreichen, müssen mindestens **54 Punkte** erreicht werden.

Benotung:

- Die Lehrerin bzw. der Lehrer legt fest, ob und ggf. wie der Test benotet werden soll.

Kompetenztest **Stufe I** **Komponente A**

Handlungsvorschrift: Eine Erfindung zu Geld machen

1. Machen Sie eine Erfindung.
2. Recherchieren Sie, ob es diese Erfindung schon gibt.
3. Wenn es die Erfindung schon gibt, gehen Sie zu 1.
4. Reichen Sie einen Antrag für ein Patent über die neue Erfindung beim Patentamt ein.
5. Bezahlen Sie die Gebühren.
6. Wenn Sie eine zahlungskräftige Firma finden, die Interesse an der Erfindung hat, dann verkaufen Sie das Patent an diese Firma.
7. Wenn Sie keine zahlungskräftige Firma finden, die Interesse an der Erfindung hat, dann gründen Sie eine eigene Firma und entwickeln Sie ein Produkt, das sich gut verkauft.

Erfüllt die gegebene Handlungsvorschrift die Eigenschaft Allgemeinheit?

O JA

O NEIN

Erfüllt die gegebene Handlungsvorschrift die Eigenschaft Eindeutigkeit?

O JA

O NEIN

2 Punkte

Erklären Sie den Begriff Algorithmus.

1 Punkt

Nennen Sie ein Beispiel für ein Problem, das mithilfe von Algorithmen lösbar ist.

1 Punkt

Geben Sie ein Beispiel für eine Wiederholungsanweisung (Schleife) als Pseudocode an.

1 Punkt

Erklären Sie die allgemeine Funktionsweise dieser Schleife.

1 Punkt

Geben Sie die Werte der Integervariablen a und b nach der Abarbeitung folgender Anweisungsfolge an:

Weise der Variablen *a* den Wert *1* zu.
Weise der Variablen *b* den Wert *3* zu.

Wenn die Bedingung *(a > b)* wahr ist, mache Folgendes:

Weise der Variablen *b* den Wert von *b+a* zu.
Weise der Variablen *a* den Wert *2* zu.

2 Punkte

Wenn die Bedingung *(a > b)* falsch ist, mache Folgendes:

Weise der Variablen *b* den Wert von *b-a* zu.

a →
b →

Kompetenztest **Stufe I** **Komponente C**

Marco fährt jedes Jahr zu den Geburtstagen seiner Großeltern.
Runde Geburtstage werden immer in der Gaststätte gefeiert.
Er kennt die Geburtsjahre seiner Großeltern und kann so ausrechnen, wie alt sie werden.
Im Jahr 2007 hat Marco folgenden Algorithmus zur Altersberechnung entwickelt.

PROGRAMM Altersberechnung

```
Lege die Variablen geburtsjahr und alter vom Typ Integer an.
Fordere den Benutzer mit "In welchem Jahr sind Sie geboren?" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable geburtsjahr.
Weise der Variablen alter den Wert von 2007-geburtsjahr zu.
Gib "Sie werden/wurden in diesem Jahr: " und danach den Wert von alter aus.
Gib den Text " Jahre alt" aus.
```

Lesen Sie den Algorithmus und übertragen Sie ihn in ein Computerprogramm. 4 Punkte

Verändern Sie das Programm so, dass der Benutzer nicht nur ein Geburtsjahr, sondern auch das aktuelle Jahr eingeben kann. Das Programm soll dann das Alter im aktuellen Jahr berechnen. 2 Punkte

Erweitern Sie das Programm so, dass bei allen Geburtstagen, bei denen das Alter durch 10 teilbar ist (dann gilt: *alter MOD 10 = 0*), ausgegeben wird: „Dies ist ein runder Geburtstag“. 2 Punkte

Kompetenztest **Stufe I** **Komponente D**

Marcos jüngere Schwester Anne lernt im Mathematikunterricht, Flächen von Rechtecken und Quadraten zu berechnen. In einer Hausaufgabe hat sie Folgendes berechnet:

Quadrat 1: Seitenlänge: 4 cm
Flächeninhalt: 16 cm²

Rechteck 1: Seitenlänge a: 5 cm
Seitenlänge b: 3 cm
Flächeninhalt: 20 cm²

Anne wünscht sich ein Programm, mit dem sie überprüfen kann, ob sie bei ihren Hausaufgaben richtig gerechnet hat.

Entwerfen und implementieren Sie das gewünschte Programm zur Berechnung des Flächeninhalts. Dabei soll der Benutzer zwischen einem Rechteck und einem Quadrat auswählen können und anschließend je nach Figur eine oder zwei Seitenlängen angeben. Das Programm soll dann den Flächeninhalt ausgeben. 6 Punkte

Testen Sie das Programm mit den Werten von Annes Hausaufgabe.
Welche Flächeninhalte gibt Ihr Programm aus? 2 Punkte

Quadrat 1: _____ Rechteck 1: _____

Kompetenztest **Stufe II** **Komponente A**

Erklären Sie die Eigenschaft Allgemeingültigkeit von Algorithmen **an einem Beispiel**.

1 Punkt

Handlungsvorschrift: Body-Mass-Index eines 20-jährigen berechnen.
Quadrieren Sie die in Metern angegebene Größe einer Person.
Berechnen Sie die Körpermassenzahl, indem Sie das in Kilogramm angegebene Gewicht der Person durch die quadrierte Größe teilen.
Ist die Körpermassenzahl zwischen 20 und 25, so hat die Person Normalgewicht.
Ist die Körpermassenzahl größer als 25, so hat die Person Übergewicht.
Ist die Körpermassenzahl kleiner als 20, so hat die Person Untergewicht.

Sind diese Eigenschaften bei der gegebenen Handlungsvorschrift erfüllt?

Eigenschaft	Endlichkeit	Eindeutigkeit	Allgemeingültigkeit	Ausführbarkeit
Erfüllt: JA / NEIN?	O JA O NEIN	O JA O NEIN	O JA O NEIN	O JA O NEIN

4 Punkte

Entscheiden Sie, ob die gegebene Handlungsvorschrift ein Algorithmus ist.
Begründen Sie Ihre Entscheidung.

1 Punkt

Nennen Sie zwei Beispiele für Probleme, die mithilfe von Algorithmen **nicht** lösbar sind.

Beispiel 1:

Beispiel 2:

2 Punkte

Geben Sie ein Beispiel für eine Wiederholung mit einer Zählschleife (FOR-Schleife) als Pseudocode und als Struktogramm oder Programmablaufplan an.

Pseudocode	Struktogramm oder Programmablaufplan

2 Punkte

Welche Ausgaben liefert das folgende Programmfragment?

Weise der Variablen *a* den Wert *14* zu.
Weise der Variablen *b* den Wert *3* zu.
Weise der Variablen *c* den Wert von *a DIV b* zu.
Weise der Variablen *d* den Wert von *a MOD b* zu.
Gib den Wert von *c* aus.
Gib " und " und danach den Wert von *d* aus.
Gib "/" und danach den Wert von *b* aus.

1 Punkt

Ausgabe:

Nennen Sie zwei Datentypen.

2 Punkte

Geben Sie für den folgenden Satz einen entsprechenden Ausdruck einer Programmiersprache an:
Der Wert der Variablen a soll größer als 5 und kleiner als 25 sein.

1 Punkt

Erklären Sie den Unterschied zwischen WHILE-DO-Schleife und REPEAT-UNTIL-Schleife.

2 Punkte

Kompetenztest **Stufe II** **Komponente C**

Marco hat im Geografieunterricht die Aufgabe bekommen, für verschiedene Städte die minimale und die maximale Tagesdurchschnittstemperatur dieser Woche anzugeben. Nach einer Recherche im Internet hat er die Temperaturwerte für alle Städte und Wochentage zusammengetragen. Das Heraussuchen der Minimum- und der Maximumtemperatur möchte Marco mithilfe eines Computerprogramms erledigen. Marco hat sich den folgenden Algorithmus zur Berechnung der minimalen Wochentemperatur überlegt:

PROGRAMM Minimumtemperatur

Lege die Variablen *tag*, *min* und *temp* vom Typ Integer an.
Fordere den Benutzer mit "Bitte Temperatur eingeben:" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable *temp*.
Weise der Variablen *min* den Wert von *temp* zu.

Lasse den Wert der Variablen *tag* von *2* bis *7* in Schritten der Größe *1* laufen und mache Folgendes:

> Fordere den Benutzer mit "Bitte Temperatur eingeben:" auf, eine Zahl einzugeben.
> Speichere die eingegebene Zahl in die Variable *temp*.
>
> Wenn die Bedingung *(temp < min)* wahr ist, mache Folgendes:
>
> > Weise der Variablen *min* den Wert von *temp* zu.

Gib "Die Minimumtemperatur: " und danach den Wert von *min* aus.

Lesen Sie den gegebenen Algorithmus und geben Sie seine Funktionsweise mit eigenen Worten wieder.

4 Punkte

Für London hat Marco folgende Temperaturen herausgesucht:
Montag 8°, Dienstag 11°, Mittwoch 7°, Donnerstag 9°, Freitag 11°, Samstag 8°, Sonntag 10°

Entwickeln Sie zu dem gegebenen Algorithmus eine Durchlauftabelle für die Temperaturen in London.

tag	min	temp

3 Punkte

Setzen Sie den gegebenen Algorithmus in ein Programm um. 6 Punkte

Erweitern Sie das Programm so, dass die Minimum- und die Maximumtemperatur der Woche ausgegeben werden. 3 Punkte

Kompetenztest **Stufe II** **Komponente D**

Marco hat im letzten Monat zu viel telefoniert. Seine Eltern haben jetzt beschlossen, ihm einen festen Betrag für Telefonate bereitzustellen. Marco will nun günstiger telefonieren. Er hat die Minutenpreise von vier unterschiedlichen Anbietern und die zugehörigen Telefonnummern herausgesucht.

	Hauptzeit Festnetz	Nebenzeit Festnetz	Hauptzeit Mobil	Nebenzeit Mobil
Telefonus (01234)	0,03	0,02	0,38	0,30
Mobilus (09876)	0,07	0,05	0,25	0,20
Genialion (0547490)	0,04	0,01	0,32	0,27
Billigus (0150705)	0,05	0,03	0,27	0,18

Kreuzen Sie den jeweils günstigsten Anbieter in jeder der vier Spalten an.

Fertigen Sie einen schriftlichen Entwurf für ein Programm an, bei dem der Benutzer zwischen Haupt- und Nebenzeit sowie zwischen Festnetz- und Mobilnummer auswählen kann. Anschließend soll der günstigste der vier gegebenen Anbieter mit Nummer ausgegeben werden.

5 Punkte

Implementieren Sie das entworfene Programm benutzungsfreundlich. 6 Punkte

Testen Sie Ihr Programm mit folgenden Eingaben:

A) ein Telefonat zu einer Festnetznummer in der Nebenzeit
B) ein Telefonat zu einer Mobilfunknummer in der Hauptzeit

Notieren Sie alle Eingaben und Ausgaben Ihres Programms bei beiden Testfällen.

4 Punkte

Reflektieren Sie über Ihren Lösungsweg. Welche Schwierigkeiten traten beim Entwurf und bei der Implementierung auf? Wie haben Sie diese Probleme gelöst?

1 Punkt

Nennen Sie ein Beispiel für einen Algorithmus, der beim Benutzen eines Mobiltelefons auftritt.

1 Punkt

Erklären Sie **an dem von Ihnen genannten Beispiel** die vier Eigenschaften von Algorithmen.

Endlichkeit
Ausführbarkeit
Allgemeingültigkeit
Eindeutigkeit

8 Punkte

Geben Sie Beispiele für Handlungsvorschriften an, die die Eigenschaften Eindeutigkeit, Ausführbarkeit und Allgemeingültigkeit **nicht** erfüllen.

Geben Sie eine Handlungsvorschrift an, die **nicht eindeutig** ist.	
Geben Sie eine Handlungsvorschrift an, die **nicht ausführbar** ist.	
Geben Sie eine Handlungsvorschrift an, die **nicht allgemeingültig** ist.	

3 Punkte

Geben Sie ein Beispiel für einen einfachen Algorithmus mit Eingabe, Verarbeitung und Ausgabe als Pseudocode an und überführen Sie ihn in ein Struktogramm oder einen Programmablaufplan.

Pseudocode	Struktogramm oder Programmablaufplan

6 Punkte

Erklären Sie die folgenden Begriffe:

Prozedur:

1 Punkt

Parameter:

1 Punkt

Was gibt das folgende Programm aus?

Prozedur *test*(Referenzparameter *a* vom Typ Integer, Wertparameter *b* vom Typ Boolean)

```
Weise der Variablen a den Wert von a+2 zu.
Weise der Variablen b den Wert FALSE zu.
Gib den Wert von a aus.
Gib den Wert von b aus.
```

Hauptprogramm

```
Lege die Variable r vom Typ Integer an.
Lege die Variable s vom Typ Boolean an.
Weise der Variablen r den Wert 7 zu.
Weise der Variablen s den Wert TRUE zu.
Rufe die Prozedur test mit den Parametern r, s auf.
Gib den Wert von r aus.
Gib den Wert von s aus.
```

Ausgabe:

4 Punkte

Kompetenztest **Stufe III** **Komponente C**

Marco hat im Mathematikunterricht das Rechnen mit Brüchen gelernt. Bei verschiedenen Operationen muss Marco den größten gemeinsamen Teiler berechnen.
Marco nutzt für die Kontrolle seiner Hausaufgaben den folgenden Algorithmus zur Berechnung des größten gemeinsamen Teilers (ggT):

```
PROGRAMM ggT
Lege die Variablen a und b vom Typ Boolean an.
Fordere den Benutzer mit "1. Zahl eingeben:" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable a.
Fordere den Benutzer mit "2. Zahl eingeben:" auf, eine Zahl einzugeben.
Speichere die eingegebene Zahl in die Variable b.

Solange wie (a # b) wahr ist, mache Folgendes:
    Wenn die Bedingung (a > b) wahr ist, mache Folgendes:
        Weise der Variablen a den Wert von a-b zu.
    Wenn die Bedingung (a > b) falsch ist, mache Folgendes:
        Weise der Variablen b den Wert von b-a zu.

Gib "Der ggT der beiden Zahlen ist: " und danach den Wert von a aus.
```

Lesen Sie den Algorithmus und geben Sie seine Funktionsweise mit eigenen Worten wieder.

4 Punkte

Der gegebene Pseudoquelltext enthält einen Fehler. Kreuzen Sie die fehlerhafte Zeile im Algorithmus an und schreiben Sie auf, wie diese richtig heißen müsste. 1 Punkt

Überprüfen Sie den korrigierten Algorithmus für ein typisches und ein untypisches Beispiel mithilfe von Durchlauftabellen.

6 Punkte

Setzen Sie den Algorithmus in ein Programm um. 6 Punkte

Verändern Sie das Programm so, dass das Berechnen des größten gemeinsamen Teilers in einem Unterprogramm erfolgt. Eingaben und Ausgaben sollen im Hauptprogramm bleiben. 3 Punkte

Verändern Sie das Programm so, dass der größte gemeinsame Teiler von drei Zahlen berechnet werden kann. 4 Punkte
Greifen Sie dabei auf das Unterprogramm zur Berechnung des größten gemeinsamen Teilers von zwei Zahlen zurück. ggt (a , b , c) = ggt (ggt (a , b) , c)

Kompetenztest **Stufe III** **Komponente D**

Marcos Schwester Anne hat in der Schule das Nim-Spiel gelernt.
Das Nim-Spiel ist ein Spiel für zwei Personen:

Gegeben sind 13 Streichhölzer.
Abwechselnd nimmt jeder Spieler je ein, zwei oder drei Streichhölzer.
Derjenige, der das letzte Streichholz wegnimmt, hat gewonnen.

Anne spielt das Nim-Spiel mit ihrem Bruder Marco.
Anne verliert immer, wenn Marco anfängt, zu ziehen.
Sie hat die Vermutung, dass Marco betrügt.

Anne wünscht sich ein Programm, bei dem zwei Menschen am Computer gegeneinander spielen können.
Das Programm soll überprüfen, ob die Regeln von beiden Spielern eingehalten werden. Über eine Eingabe sollen die Benutzer bei jedem Spielzug entscheiden können, ob ein, zwei oder drei Streichhölzer vom Stapel genommen werden sollen.
Anschließend soll angezeigt werden, wie viele Streichhölzer noch auf dem Stapel sind.
Falsche Eingaben sollen nicht akzeptiert werden.
Am Ende des Spiels soll ausgegeben werden, welcher Spieler gewonnen hat.

Fertigen Sie einen schriftlichen Entwurf für das gewünschte Programm an.

6 Punkte

Implementieren Sie das Programm benutzungsfreundlich. 8 Punkte

Testen Sie das Programm an selbst gewählten Eingaben.
Notieren Sie Eingaben und Ausgaben Ihres Tests.

2 Punkte

Diskutieren Sie Vor- und Nachteile der Computerlösung.

4 Punkte

Verbessern oder erweitern Sie das Programm in sinnvoller Weise. 4 Punkte