21. Bundeswettbewerb Informatik 2002/2003, 1. Runde

Lösungshinweise und Bewertungskriterien

Allgemeines

Zuerst soll an dieser Stelle gesagt sein, dass wir uns sehr darüber gefreut haben, dass einmal mehr so viele Leute sich die Mühe gemacht und die Zeit zur Bearbeitung der Aufgaben genommen haben. Dass das Zeitnehmen nicht immer optimal klappt und es deshalb kurz vor Einsendeschluss etwas brenzlig wird, ist nachvollziehbar und völlig verständlich. Leider dürfen wir darauf keine Rücksicht nehmen. Insbesondere sind vollständige Einsendungen ein Muss. Im Einzelnen:

- Beinahe das Wichtigste ist, die Teilnahmeformulare vollständig und leserlich auszufüllen und für jedes Gruppenmitglied auch wirklich ein eigenes, vollständig ausgefülltes Formular abzugeben. Zu unnötig langen Suchprozessen führt es, wenn die Formulare zwischen den Aufgaben versteckt sind oder Teilnehmer ihre Namen nicht zumindest auf die erste Seite der Lösung schreiben – und zwar bei jeder Aufgabe!
- Online-Anmelder wurden gebeten, ihre Nummer außen auf den Umschlag zu schreiben. Die meisten haben das gemacht, aber leider nicht alle. Das führt zu Komplikationen und verzögert insbesondere die versprochene Rückmeldung per E-mail über den Eingang der Einsendung.
- Was uns auch hilft: Seien Sie mit Ihrem Namen nicht so geizig! Schreiben Sie Ihn ruhig häufiger, z. B. auf das erste Blatt jeder Aufgabe und auf Ihre Diskette(n) oder CD.
- Beispiele werden als Teile des Programmablauf-Protokolls immer erwartet. In diesem Jahr waren bei den meisten Aufgaben mehrere Beispiele gefordert und bei allen Aufgaben ein Beispiel vorgegeben. Zu wenig Beispiele und erst recht die Nichtbearbeitung des vorgegebenen Beispiels führten zu Punktabzug.
- Beispiele, aber auch Programmdokumentation und Programmtext müssen ausgedruckt sein. Wir können aus Zeit- und Kostengründen keine Ausdrucke machen, so dass es prinzipiell nichts nützt, Beispiele nur auf Diskette/CD abzugeben oder gar nur ins Programm einzubauen. In solchen Fällen gab es Punktabzug.

- Noch schlechter als Einsendungen nur auf Datenträgern wären für uns übrigens Einsendungen via E-mail oder anderen Internet-Wegen, auch wenn das für die Teilnehmer noch so praktisch wäre. Papiereinsendungen sind (zumindest zur Zeit und sicher auch noch in den nächsten Jahren) einfach unumgänglich.

- Eine Gruppeneinsendung schicken Sie uns bitte in einem gemeinsamen Umschlag, wir haben sonst größte Mühe, die Einsendungen richtig einzusortieren. Wenn mehrere Einsendungen in einen Umschlag gesteckt werden, geben Sie bitte bei der Online-Anmeldung bzw. auf den Anmeldebögen bitte die Zusammensetzung Ihrer Gruppe an. Außerdem: Bei Gruppeneinsendungen müssen Sie sich auf jeweils eine Lösung pro Aufgabe einigen, und Gruppenmitglieder können nicht gleichzeitig auch eine eigene Einsendung schicken.

So, vielleicht nehmen Sie sich diese Anmerkungen ja zu Herzen, wenn Sie (hoffentlich) im nächsten Jahr wieder mitmachen.

Auch die folgenden eher inhaltlichen Dinge sollten Sie beachten:

- Lösungsideen sollten Lösungsideen sein und keine Bedienungsanleitungen oder Wiederholungen der Aufgabenstellung. Es soll beschrieben werden, welches Problem hinter der Aufgabe steckt und wie dieses Problem grundsätzlich angegangen wird. Eine einfache Mindestbedingung: Bezeichner von Programmelementen wie Variablen, Prozeduren etc. dürfen nicht verwendet werden – eine Lösungsidee ist nämlich unabhängig von solchen Realisierungsdetails. In diesem Jahr waren die meisten Lösungsideen in dieser Hinsicht recht gut, oft aber ein wenig kurz.

- Auch ein Programmablauf-Protokoll soll keine Bedienungsanleitung sein. Es beschreibt nicht, wie das Programm ablaufen sollte, auch nicht die zum Ablauf nötigen Interaktionen mit dem Programm, sondern protokolliert den tatsächlichen, inneren Ablauf eines Programms. Sprich: das Programm protokolliert seinen Ablauf selbst, z. B. durch Herausschreiben von Eingaben, Zwischenschritten oder -resultaten und Ausgaben.

- Oben wurde schon gesagt, dass Beispiele immer dabei sein sollten, zumindest eines davon in einem ausführlichen Programmablauf-Protokoll. Das hat natürlich seinen Grund: An den Beispielen ist oft direkt zu sehen, ob bestimmte Punkte korrekt beachtet wurden. Viele meinen nun, wir könnten die Programme ja laufen lassen und selbst auf Beispieldaten ansetzen, und liefern keine Beispiele oder nur Beispieldaten in elektronischer Form. Das können wir aber in der Regel nicht, weil die Zeit dazu viel zu knapp ist. Außerdem ist nicht immer sicher, dass Programme, die auf dem eigenen PC laufen, auch auf einem anderen Computer ausführbar sind; das kann verschiedenste Gründe haben. Generell muss man sich darauf einstellen, dass eventuell nur das Papiermaterial angesehen wird!

- Mit den verschiedenen Beispielen sollten sie wichtige Varianten des Programmablaufs zeigen, also auch Sonderfälle, die vom Programm berücksichtigt werden.

Einige Anmerkungen noch zur Bewertung:

- Auf den Bewertungsbögen bedeutet ein Kreuz in einer Zeile, dass die (negative) Aussage in dieser Zeile auf Ihre Einsendung zutrifft. Damit verbunden war dann in der Regel der Abzug eines oder mehrerer Punkte. Eine Wellenlinie bedeutet „na ja, hätte besser sein können“, führte aber meist nicht zu Punktabzug. Mehrere Wellenlinien konnten sich aber zu einem Punktabzug addieren.
- Wellenlinien wurden übrigens häufig für die Dokumentation (also Lösungsidee, Programm-Dokumentation, Programmablauf-Protokoll und kommentierter Programm-Text) verteilt, obwohl Punktabzug auch gerechtfertigt gewesen wäre.
- Aber auch so ließ sich nicht verhindern, dass etliche Teilnehmer nicht weitergekommen sind, die nur drei Aufgaben abgegeben haben in der Hoffnung, dass schon alle richtig sein würden. Das ist ziemlich riskant, da Fehler sich schnell einmal einschleichen.

Zum Schluss:

- Sollte Ihr Name auf der Urkunde falsch geschrieben sein, können Sie gerne eine neue anfordern. Uns passieren durchaus schon mal Tippfehler, und gelegentlich scheitern wir an der ein oder anderen Handschrift.
- Es ist verständlich, wenn jemand, der nicht weitergekommen ist, über eine Reklamation nachdenkt. Gehen Sie aber bitte davon aus, dass wir in allen kritischen Fällen, insbesondere also denen mit 11 Punkten, ganz genau nachgeprüft haben, ob nicht doch noch irgendwoher die Punkte zu holen gewesen wären, die das Weiterkommen ermöglicht hätten.

Wir wollen auch in diesem Jahr zusätzlich zu diesen Lösungsideen ausführlichere Beispiellösungen erstellen, die auf den Webseiten des Bundeswettbewerbs Informatik (`www.bwinf.de`) veröffentlicht werden – allerdings wohl nicht vor Ende Januar. Bis dahin kann man sich mit den Einsendungen befassen, die unter `www.tobias-thierer.de/bwifiles.html` von einigen Teilnehmern veröffentlicht wurden.

Aufgabe 1: Graf Rüdeger

Lösungsidee

Beim Betrachten der vier Regeln zum Verschieben der Riegel fällt Folgendes auf: Um einen beliebigen Riegel x bewegen zu können, ist lediglich die Stellung der Riegel mit Nummern kleiner x von Bedeutung. Diese müssen in eine bestimmte Stellung entsprechend der Regeln 1 bis 3 gebracht werden, bevor Riegel x bewegt werden kann. Um bereits erreichte Endstellungen nicht wieder „kaputt“ zu machen, müssen also zuerst die Riegel mit hohen Nummern in die Endstellung gebracht werden, bevor dann die Riegel mit kleineren Nummern in ihre Zielstellung gebracht werden können.

Um die Pforte ordentlich zu schließen, muß man demnach zuerst Riegel N in die richtige Position bringen. Wenn man Glück hat, so ist Riegel N bereits geschlossen. Hat man Pech und Riegel N ist geöffnet, so müssen zuerst die Riegel $N-1$ bis 1 in die benötigte Stellung entsprechend Regel 3 bzw. 2 gebracht werden, bevor Riegel N endgültig geschlossen werden kann. Auf die gleiche Weise schließt man danach (rekursiv) der Reihe nach Riegel $N-1$ bis Riegel 1.

Es bleibt nun noch zu klären, wie man die Riegel mit den kleineren Nummern in die benötigen „Zwischenpositionen“ bekommen. Dazu lässt sich die obige Strategie verallgemeinern und für $m \leq N$ so beschreiben:

```
Bringe Riegel in Stellung s_m,...,s_1:
    Wenn Riegel m nicht in Stellung s_m dann
        t_{m-1},...,t_1 sei Stellung um Riegel m bewegen zu können
        call „Bringe Riegel in Stellung t_{m-1},...,t_1"
        Bewege Riegel m
    end
    call „Bringe Riegel in Stellung s_{m-1},...,s_1"
end
```

Um eine von Graf Rüdegers Schließanlagen ordentlich zu schließen, muss die obige Prozedur nun nur noch mit dem gewünschten Endzustand aufgerufen werden:

```
call „Bringe Riegel in Stellung zu,...,zu"
```

Alternativ zum rekursiven Ansatz ist auch ein gleichwertiges iteratives Vorgehen möglich. Hierbei wird von Riegel N ausgehend versucht, die Riegel zu schließen. Ist das für einen Riegel m nicht möglich, wird von dort aus so lange wie nötig der nächstliegende bewegliche Riegel bewegt – unter der Einschränkung, dass ein Riegel nicht zweimal hintereinander bewegt werden darf, um endloses Öffnen und Schließen desselben Riegels zu vermeiden.

Verwendung von Regel 4

Regel 4 ist, genau genommen, überflüssig. Sie dient lediglich dem Bewegen von Riegel N, der aber bereits mit Regel 3 (für $N > 2$) bzw. Regel 2 ($N = 2$) oder Regel 1 ($N = 1$) bewegt werden kann. Man kann also auf das Anwenden dieser Regel verzichten. Dadurch vereinfacht sich das Programm, da es nun nur noch drei Regeln zu beachten gibt. Regel 4 erlaubt aber das Abkürzen für den speziellen Fall, daß alle Riegel bis auf Riegel N bereits geschlossen sind. Diese Abkürzung ist nützlich, wenn man die Pforte in den ordentlich geschlossenen Zustand überführen möchte. Und da genau das in der Aufgabenstellung gefordert wird, ist die Verwendung von Regel 4 sinnvoll: Zuerst werden die Riegel $N-1$ bis 1 mittels der oben vorgestellten Strategie geschlossen und im letzten Schritt (falls notwendig) Riegel N mittels Regel 4.

Pflichtbeispiel

Um eine Schließanlage mit 6 Riegeln ($N = 6$), die sich im Zustand (zu, zu, zu, auf, auf, zu) befindet, werden folgende Bewegungen benötigt:

$$5, 1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, 3, 1, 2, 1.$$

Regel 4 wird nicht in Anspruch genommen, da Riegel $N = 6$ schon von vornherein zu ist.

Bewertungskriterien

Bei der Bewertung wurden folgende Punkte besonders beachtet:

- Es muss diskutiert werden, in welcher Folge die Riegel ausgewählt werden (denn das ist der Kern jeder Lösungsidee).
- Die besondere Rolle von Regel 4 muss erkannt worden sein.
- Das rekursive Prinzip bzw. das Prinzip, von Riegel N auszugehen, das hinter den Regeln 1 bis 3 steckt (vgl. Türme von Hanoi!) muss erkannt worden sein.

Aufgabe 2: Reservierungssystem

Lösungsidee

Aufgabenteil 1 – Grobkonzeption

Das Reservierungssystem sollte Folgendes leisten:

1. Entgegennahme aller erforderlichen Daten zur Reservierung selbst (Zeitraum; Hotel, Pension oder Ferienhaus; Anzahl Zimmer bzw. Personen) und zur Person des Anrufers (Anschrift, Email-Adresse). Bei unzureichendem Kenntnisstand muss das System nachfragen. Bei Verständnisschwierigkeiten wird an einen menschlichen Bearbeiter durchgestellt.
2. Beantwortung der Reservierungsanfrage mit Vorschlägen zur Unterkunft inkl. Informationen über Ausstattung, Service und Preise.
3. Schnittstelle zu Hotelbetreibern und Vermietern (evtl. auch sprachgesteuert), um diese über die gebuchten Reservierungen in Kenntnis zu setzen und ihnen die Aufnahme oder Änderung ihres Angebots zu ermöglichen.

Ein solches System besteht grob aus folgenden Komponenten:

Spracherkennung und -ausgabe Umwandlung des Telefongesprächs in Text und (für Antworten) umgekehrt.

Dialogsystem Analysiert die Äußerungen des Anrufers, wandelt sie in ein internes Format um und generiert eigene Äußerungen, wobei es bestimmte Zwecke verfolgt. In einer ersten Phase ermittelt das System die Reservierungsdaten. Wenn diese bekannt sind, kann es in einer zweiten Phase mit Hilfe von Anfragen an die Buchungsdatenbank (s.u.) passende Unterkünfte oder auch Alternativen präsentieren. Generell ist eine Aufteilung dieser Komponente in Sprachanalyse, Sprachgenerierung und Dialogsteuerung sinnvoll.

Buchungsdatenbank Enthält sämtliche touristischen Angebote von Bitshafen inkl. ihres Reservierungsstands. Das System kann so berücksichtigen, dass Angebote schon ausgebucht sein könnten.

Schnittstelle zu den Tourismus-Anbietern Schnittstelle zur Buchungsdatenbank, über die Anbieter ihre Angebote bearbeiten und sich über Buchungen informieren können.

Aufgabenteil 2 – Reservierungsannahme

In Teil 2 soll das Dialogsystem in seiner ersten Phase implementiert werden. Die Beispiele geben die zu realisierende Leistungsfähigkeit vor und sind damit auch Testfälle. Um diese Testfälle zu bewältigen, ist eine schematische Musteranalyse ausreichend. Sie kann sich darauf beschränken,

gewisse Wörter und Wortmuster in der Äußerung zu suchen, die die benötigten Informationen enthalten.

Wichtig ist die Organisation der Musteranalyse. Für diese Aufgabe ist es sinnvoll, für die verschiedenen zu ermittelnden Angaben „Spezialisten“ zu implementieren, die jeweils Suchmuster und passende Fragen zusammenfassen. Auf den ersten Satz eines Anrufers können dann alle Spezialisten mit ihren Suchmustern losgelassen werden. Danach werden für die noch fehlenden Angaben die Spezialisten mit ihren Fragen der Reihe nach aktiviert.

Folgende Daten müssen erkannt werden (für die es nach der obigen Idee Spezialisten geben kann):

1. Zeitraum in der Form *von/vom* `Tag.Monat.Jahr` *bis [zum]* `Tag.Monat.Jahr`. Dabei reicht es aus, dass das Jahr nur einmal genannt wird oder *in diesem Jahr/dieses Jahr* in der Eingabe vorkommt. Ansonsten wird das Jahr extra nachgefragt. Reservierungen über den Jahreswechsel kommen in den Beispielen nicht vor, müssen also nicht unbedingt möglich sein. Alle Jahreszahlen sollten vierstellig angegeben werden (wir wollen ja schließlich Y2K-kompatibel sein).

2. Typ der Unterkunft: entweder Ferienhaus, Einzelzimmer oder Doppelzimmer (auch in der Form *Zimmer für* `n` *Personen*). Wird ein Zimmer für mehr als zwei Personen verlangt, schlägt das Programm ein Ferienhaus vor. Zusätzlich wird auch gleich noch die Anzahl Zimmer bzw. Personen erfasst, falls schon mit angegeben.

3. Anzahl der Zimmer/Komfort (nur beim Fall Zimmer): Falls die Anzahl der Zimmer noch nicht feststeht, wird hier noch einmal nachgefragt und danach noch die Frage geklärt, ob es eine Pension oder ein Hotel sein soll.

4. Anzahl der Personen/Erwachsenen/Kinder (nur beim Fall Ferienhaus): Gibt der Anrufer spezifizierte Zahlen für Kinder und/oder Erwachsene an, so wird das auch gespeichert und zusätzlich die Gesamtzahl ausgegeben. Ansonsten wird nur nach der Gesamtzahl nachgefragt, allerdings auch eine erweiterte Antwort voll ausgewertet.

Aufgabenteil 3 – Konsequenzen eines elektronischen Reservierungssystems

Durch die Einführung eines elektronischen Reservierungssystems ließen sich sicherlich Kosten für die Verwaltung sparen. Diese Einsparungen könnten an die Anbieter von Unterkünften weitergegeben oder vielleicht die Kurtaxe gesenkt werden.

Für die nachfragenden Touristen ergibt sich eine zwiespältige Situation: Einerseits könnte für sie der Anruf billiger (vielleicht jetzt eine kostenlose Nummer) und bequemer werden (24h-Service), andererseits gibt es auch viele Nachteile: Technische Probleme mit der Verständigung könnten zu Frustration führen. Außerdem erwartet man doch gerade in einem Touristenort eine persönliche, freundliche und zuvorkommende Behandlung. Persönliche Empfehlungen und weitergehende Informationen sind vielleicht schwieriger oder gar nicht mehr zu erhalten.

Für die weiterhin Beschäftigten ergibt sich wahrscheinlich eine interessantere Tätigkeit. Die Routine-Fälle werden Ihnen abgenommen, sie haben so mehr Zeit für die schwierigeren Anfragen. Die nicht mehr benötigten Telefonist(inn)en müssen sich einen neuen Job suchen, was zumindest derzeit nicht einfach ist.

Pflichtbeispiele

Pflichtbeispiele sind bei dieser Aufgabe nur indirekt vorgegeben. Eine Einsendung sollte nachweisen, dass das realisierte System mit den Beispielen der Aufgabenstellung klarkommt. Dieser Nachweis ist am direktesten zu führen, indem man die ersten Äußerungen aus den Beispielen des Aufgabenblattes dem eigenen System „vorwirft“. Es sollte damit nicht wortwörtlich, aber in ähnlicher Weise umgehen wie das virtuelle System der Aufgabenstellung. Wie immer werden drei Beispiele erwartet, das können die Beispiele der Aufgabenstellung sein. Die Leistungsfähigkeit der Sprachanalyse ließ sich bei der Bewertung allerdings deutlich besser abschätzen, wenn Beispiele bzw. die Reaktionen darauf von den Vorgaben der Aufgabenstellung leicht abwichen.

Bewertungskriterien

Die Aufgabe besteht aus drei Einzelteilen, wobei nur Teil 2 eine Implementationsleistung fordert. Die schriftlichen Auslassungen zu den Teilen 1 und 3 müssen nicht allzu ausführlich sein, sondern auf die gefragten Punkte vernünftig und plausibel eingehen.

Bei der Bewertung wurden entsprechend folgende Punkte besonders beachtet:

Teil 1: Die Aufgabenstellung ist hier eindeutig, wobei die Aufgabe, ein passendes Grobdesign zu erstellen, durch die darauffolgende Anweisung präzisiert wird.

- Die Beschreibung der Leistungsfähigkeit sollte vorhanden sein.
- Das Grobdesign sollte die Sprachanalyse inklusive Dialogsteuerung und die Datenbank enthalten.

Teil 2: Hier gibt es diverse kleinere Punkte, auf die zu achten ist. Das System sollte

- die in den Beispielen verwendeten Formate für Datumsangaben beherrschen (Monate können als Ordinalzahlen oder mit Namen angegeben sein). Spezialitäten wie Reservierungen über den Jahreswechsel müssen nicht berücksichtigt sein.
- die in den Aussagen enthaltenen Informationen auch erkennen können. Folglich sollte es keine Fragen stellen, die schon beantwortet wurden.

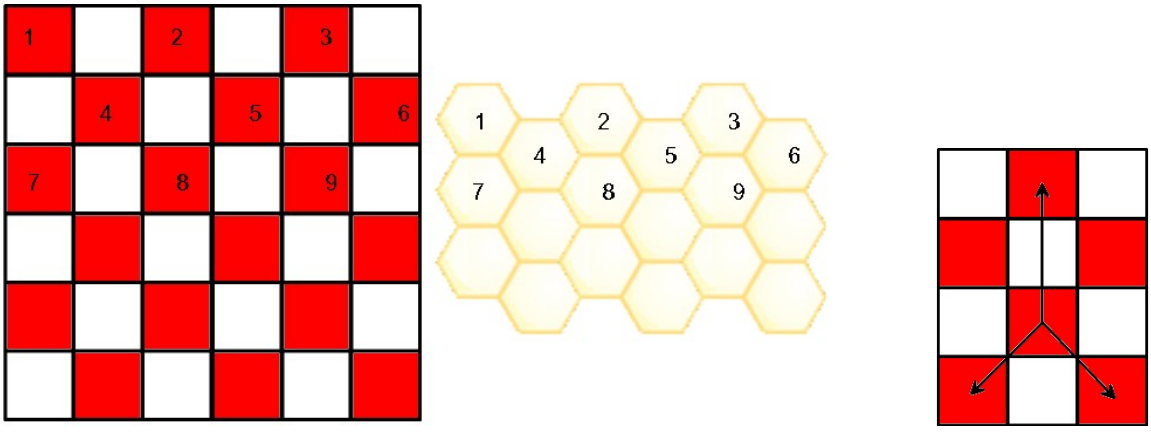
In der Aufgabenstellung wird gefordert, dass das System Angaben zur Zahl von Erwachsenen und Kindern entgegennehmen kann. Dieser Punkt wurde bei der Bewertung ignoriert.

Teil 3: Zu diesem Teil sollte etwas halbwegs Vernünftiges geschrieben worden sein, wobei die in der Aufgabenstellung angesprochenen Personengruppen mehrheitlich erwähnt sein sollten.

Aufgabe 3: Betty Bee

Lösungsidee

Die eigentliche Schwierigkeit bei dieser Aufgabe besteht darin, eine geeignete Datenstruktur zur Wiedergabe der Waben zu finden. Einige wenige Teilnehmer haben hier eine Pointer-verkettete Struktur aufgebaut. Wesentlich einfacher ist es, ein zweidimensionales Array zu nehmen und nur jedes zweite Feld zu nutzen:



In dieser Array-Struktur lässt sich nun (s. Bild rechts) ein Schritt nach Norden durch (0;-2) darstellen, ein Schritt nach Westen durch (-1;+1) und ein Schritt nach Osten durch (+1;+1). Es gibt noch verschiedene andere Möglichkeiten, Waben auf zweidimensionale Felder abzubilden. Besonders gut sind die Varianten, bei denen die Bewegungen unabhängig von der Position im Array realisiert werden können. Akzeptiert haben wir auch jede andere funktionierende Struktur, wenn sie die Wegsuche nicht allzu sehr erschwerte.

Für den Algorithmus zur Wegsuche gibt es zwei nahe liegende Lösungen: Brute Force und systematischer Ausschluss. Beim Brute Force Ansatz wird für jede Zelle, die Betty besuchen kann, eine Tiefensuche gestartet, so dass letztlich alle Möglichkeiten für einen BWInF ausprobiert werden. Ziel der Tiefensuche ist, jedes Feld genau einmal zu benutzen. Es stellt sich heraus, dass die Rechenzeit für diesen Ansatz und das geforderte Beispiel im Bereich weniger Sekunden liegt. Der Ansatz ist also sogar hinreichend schnell.

Alternativ kann man einen wesentlich komplexeren Ansatz wählen, den wir in der ersten Runde aber nicht erwartet haben. Dieser Ansatz soll hier nur der Vollständigkeit halber vorgestellt werden. Hierzu werden einige allgemeine Regeln für eine Zelle z aufgestellt:

1. Ist z von keiner Nachbarzelle aus erreichbar ist, so muß z Startzelle sein.
2. Kann von z aus keine andere Zelle erreicht werden, so muß z Endzelle des Weges sein.
3. Sind für z genau zwei der möglichen Nachfolgezellen nicht mehr betretbar, so muß Bettys Weg aus z in die letzte verbleibende Zelle führen (oder in z enden).

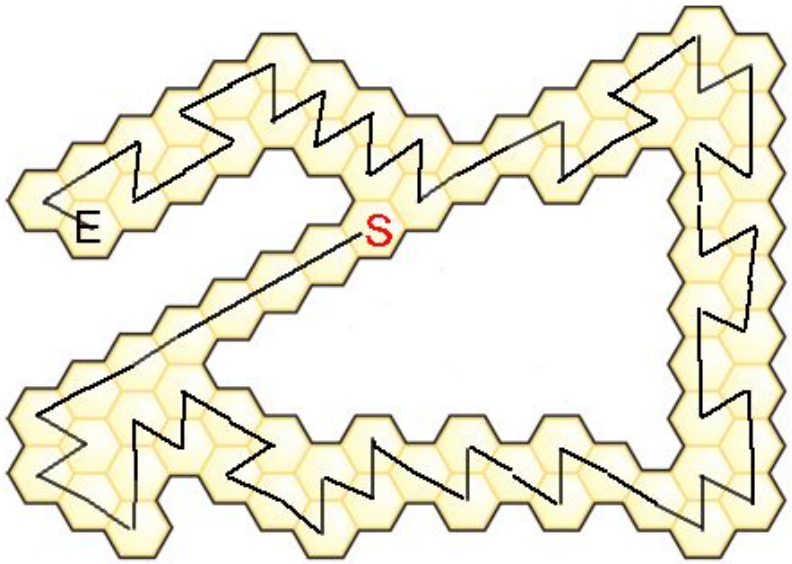
4. Kann Bettys Weg aus genau zwei der möglichen Vorgängerzellen nicht in z hineinführen, so muß der Weg aus der dritten möglichen Vorgängerzelle kommen (oder z ist Startzelle).

Mit Hilfe dieser vier Regeln lässt sich der Suchraum stark einschränken. Im geforderten Beispiel lässt sich die Lösung alleine unter wiederholtem Anwenden dieser Regeln finden. Bei vielen Waben kann mit dieser Methode alleine keine Lösung gefunden werden. Es kann lediglich der Suchraum eingeschränkt werden, so dass später effizient im verbleibenden Suchraum mit Brute Force gesucht werden kann.

Wabenstrukturen ohne möglichen Betty-Pfad lassen sich daran erkennen, dass es mehrere Start- oder Endzellen gibt oder dass es mehrere Zyklen gibt, die sich nicht mehr auflösen lassen.

Pflichtbeispiel

Für die vorgegebene Wabe gibt es genau einen BWInF:



Die zugehörige Richtungsfolge lautet:
WWWWWWOWOONNONOOWOONONOONOONOOONO
NNWNNONNWNNONNWNWWOWWNWWWNWNWNWWOWWNWWO

Bewertungskriterien

Für den Kern dieser Aufgabe, nämlich den Algorithmus zum Bestimmen eines BWInF, gab es keinerlei Einschränkungen. Selbst reine Brute Force Ansätze wurden akzeptiert, angesichts des Hinweises auf die mögliche Berechnungsdauer in der Aufgabenstellung.

Bei der Bewertung wurden folgende Punkte besonders beachtet:

- Die Datenstruktur für die Darstellung der Waben sollte geeignet sein. Es muss nicht unbedingt ein Array in der beschriebenen Art verwendet werden, aber die gewählte Datenstruktur sollte die Wegfindung nicht allzu sehr verkomplizieren.
- Die Einschränkung der Richtungen sollte korrekt umgesetzt sein.
- Jedes Feld darf nur einmal besucht werden – also müssen auch Beginn- und Endzelle eines BWInF unterschiedlich sein.
- Die Lösungen müssen in der Dokumentation grafisch dargestellt sein. Sinnvoll ist die Abbildung der bearbeiteten Wabe, in die der gefundene Weg mit Start- und Endpunkt eingezeichnet ist. Die grafische Darstellung muss nicht vom Programm erzeugt worden sein.

Aufgabe 4: Tanker im Zuse-Kanal

Lösungsidee

Diese Aufgabe kann als Graphenproblem gesehen werden. Für jedes Planquadrat der Karte in Kombination mit einer Tankergeschwindigkeit hat der Graph einen Knoten. Eine (ungerichtete) Kante existiert zwischen einem Knotenpaar (a,b) genau dann, wenn ein Tanker innerhalb einer Minute von Zustand a in Zustand b (und umgekehrt) wechseln kann, wenn also die Strecke zwischen den zugehörigen Planquadratmittelpunkten kein belegtes Planquadrat überschneidet und sich außerdem wegen des begrenzten Abbrems- und Beschleunigungsvermögens des Tankers die zugehörigen Geschwindigkeiten in keiner Koordinate um mehr als 1 unterscheiden.

Das Problem besteht dann darin, den kürzesten Weg zwischen den beiden Knoten zu finden, die zum Start- und Zielplanquadrat mit jeweils Geschwindigkeit 0 gehören. Damit ist auch die Frage aus der Aufgabenstellung beantwortet, was für den Tanker am Start und Ziel gelten muss. Je nach Betrachtungsweise muss er am Ziel eine Geschwindigkeit von 0 haben oder in jeder Koordinate einen Wert mit Betrag ≤ 1, also binnen einer Minute auf 0 bremsen können. Das Ziel ist natürlich schon vor dem Abbremsen erreicht, aber die Wegdauer kann auch inklusive „Bremsminute“ angegeben werden.

Beim Aufbau des oben genannten Graphen müssen zwei Bedingungen korrekt berücksichtigt werden, nämlich zum einen die Trägheit des Tankers in Form möglicher Geschwindigkeitsänderungen und zum anderen die Frage der Befahrbarkeit des direkten Weges von einem Planquadrat zu einem anderen. Der erste Punkt ist einfach, der zweite schon komplizierter. Hierzu kann man z.B. ermitteln, welche Begrenzungslinien von der Strecke zwischen den Planquadratmittelpunkten geschnitten werden und ob die betreffenden Planquadrate befahrbar sind oder nicht. Der Fall, dass die Verbindungsstrecke die Ecke eines Planquadrats schneidet, muss separat behandelt werden. Die dazu nötigen Berechnungen fallen nur dann genau aus, wenn rationale Zahlen oder in geeigneter Weise auch ganze Zahlen eingesetzt werden.

Pflichtbeispiel

Beim geforderten Beispiel braucht der Tanker 13 Minuten bis zum Ziel bzw. 14 Minuten, bis er am Ziel stillsteht. Es gibt nur diesen einen kürzesten Weg.

Eingabedatei

```
30 5
##############################
s..#.............######.....
#..#..##...##..#.....###.....z
#....##...####............###
##############################
```

Ausgabe

```
Lese Karte der Größe 30x5 ein...
Tankerkurs: (0, 3) -> (29, 2)
Obergrenze für erreichbare Geschwindigkeit: (8, 3)
Erzeuge Warteschlange der Größe 17850
Stillstand auf Ziel nach 14 Minute(n)
Schritt  0: Start bei ( 0,  3) mit Geschwindigkeit ( 0, 0)
Schritt  1: Fahrt zu  ( 1,  2) mit Geschwindigkeit ( 1,-1)
Schritt  2: Fahrt zu  ( 2,  1) mit Geschwindigkeit ( 1,-1)
Schritt  3: Fahrt zu  ( 3,  1) mit Geschwindigkeit ( 1, 0)
Schritt  4: Fahrt zu  ( 4,  2) mit Geschwindigkeit ( 1, 1)
Schritt  5: Fahrt zu  ( 6,  3) mit Geschwindigkeit ( 2, 1)
Schritt  6: Fahrt zu  ( 9,  3) mit Geschwindigkeit ( 3, 0)
Schritt  7: Fahrt zu  (13,  3) mit Geschwindigkeit ( 4, 0)
Schritt  8: Fahrt zu  (18,  2) mit Geschwindigkeit ( 5,-1)
Schritt  9: Fahrt zu  (22,  1) mit Geschwindigkeit ( 4,-1)
Schritt 10: Fahrt zu  (25,  1) mit Geschwindigkeit ( 3, 0)
Schritt 11: Fahrt zu  (27,  2) mit Geschwindigkeit ( 2, 1)
Schritt 12: Fahrt zu  (28,  2) mit Geschwindigkeit ( 1, 0)
Schritt 13: Fahrt zu  (29,  2) mit Geschwindigkeit ( 1, 0)
Lösung eindeutig: ja

##############################
s..#..5..6...7.....######.....
#1.#4.##...##..#..8..###...BCz
#.23.##...####........9..A.###
##############################

Tastendruck zum Programmende...
```

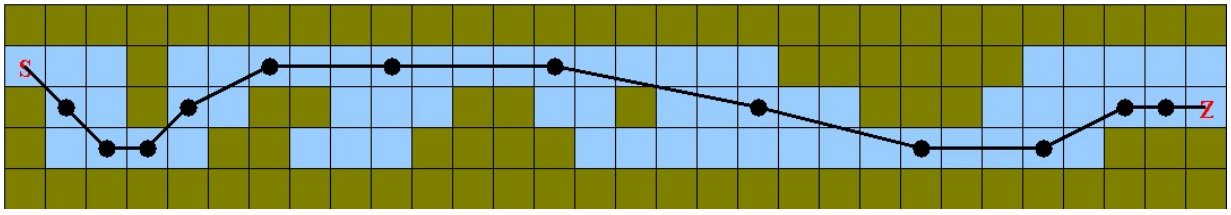


Abbildung 1: Der einzige schnellste Weg für das geforderte Beispiel

Bewertungskriterien

Der Kern dieser Aufgabe ist das Verfahren zur Bestimmung eines Fahrtweges für den Tanker. Es ist nicht gefordert, dass das Verfahren das Finden des schnellsten Weges garantiert. Sinnvolle Heuristiken, die den Tanker nach Möglichkeit in schnelle Fahrt bringen und das Finden eines existierenden Weges nicht gerade verhindern, sind akzeptabel. Bei der Bewertung wurden folgende Punkte besonders beachtet:

- Die Tankerbewegung und -trägheit muss korrekt umgesetzt sein.
- Die Prüfung der Strecken zwischen zwei Planquadraten auf Befahrbarkeit muss korrekt umgesetzt sein.
- Der Tanker muss sich in alle Richtungen bewegen können, beide Bewegungswerte müssen also auch negativ sein können.
- Gefundene Fahrten sind von akzeptabler Schnelligkeit (wenn auch nicht notwendigerweise optimal schnell). Dies ist sicher nicht der Fall, wenn der Tanker sich innerhalb einer Minute nur in ein direkt benachbartes Feld bewegen kann.
- Das Verfahren zur Wegbestimmung und seine Implementierung müssen prinzipiell auch auf andere Kanäle als den Beispielkanal anwendbar sein. Der Pflichtkanal darf also z.B. nicht im Code fest verdrahtet sein. Eine Spezifikation im Code als Wert einer Konstante bzw. Standardwert einer Variable wird gerade noch akzeptiert, falls das Programm selbst auch für andere Werte funktioniert.
- Die Lösungen müssen in der Dokumentation in geeigneter Weise grafisch dargestellt sein. Bei dieser Aufgabe genügt auch eine „ASCII-Grafik“. Die grafische Darstellung muss nicht vom Programm erzeugt worden sein.

Aufgabe 5: Grippewelle

Teilaufgabe 1: Modellierungsansatz (Lösungsidee)

Herbstzeit ist Grippezeit. Und es gibt zahlreiche Möglichkeiten, sich mit einer Grippe zu infizieren. Als Folge fühlt man sich zunächst nur etwas angeschlagen, die Nase läuft und man hat evtl. Kopfschmerzen, schließlich kommt es zu einer vollen Grippe und man bleibt am besten für ein paar Tage gleich im Bett. Dabei ist eine erkrankte Person nicht die ganze Zeit ansteckend: nicht während einer Inkubationszeit und nicht während sie isoliert im Bett liegt. Dieser zeitliche Ablauf der Infektion bei einzelnen Personen ist bei einer Simulation zu berücksichtigen – dies ist auch in der Aufgabenstellung gefordert.

Neben dem zeitlichen Ablauf ist entscheidend, in welchem Maße sich die Infektion unter den Schülerinnen ausbreitet. Hierzu kann man ein Ansteckungsmodell zu Grunde legen, das mögliche Ansteckungswege zwischen den einzelnen Personen explizit z.B. mit Hilfe eines Graphen modelliert. Die Aufgabenstellung erwartet die Modellierung von Beziehungen zwischen den Personen aber nicht. Alternativ ist möglich, einen Wert festzulegen für die Wahrscheinlichkeit, dass sich eine Schülerin an einem Tag infiziert. Das Modell wird dynamischer (und vielleicht auch realistischer), wenn dieser Wert jeden Tag anders ist und von der Anzahl der Erkrankten bzw. besser noch der am jeweiligen Tag ansteckenden Personen abhängt.

Ein solches Modell ist sicherlich nicht perfekt (und damit wären wir eigentlich schon bei Teilaufgabe 4, der Modellkritik). Es vernachlässigt völlig alle Kontakte mit anderen Personen, die sich außerhalb der Schule befinden. Auch spielen individuelle Unterschiede zwischen den Personen z.B. in Hinblick auf ihre Kontaktfreudigkeit oder ihre Konstitution bzw. den Zustand ihrer Immunsysteme keine Rolle. Eine einfache Abhilfe wäre durch die Einführung eines individuellen Faktors möglich, mit dem die generelle an einem Tag gültige Ansteckungswahrscheinlickeit zu einer individuellen Ansteckungswahrscheinlichkeit umgerechnet wird. Prinzipiell ist ein solches Modell auf andere epidemische Krankheiten übertragbar; Krankheitsablauf und Ansteckungsrate oder -graph müssen entsprechend angepasst werden.

Pflichtbeispiel

Das implementierte Modell soll für die in der Aufgabenstellung angegebenen Zahlen, nämlich für 38 Personen und eine Dauer von 70 Tagen durchgespielt werden. Weitere Beispiele sind nicht explizit verlangt. Allerdings verlangt das Aufgabenblatt generell „mehrere unterschiedliche Beispiele“. Dabei ist es sinnvoll, einen weiteren Beispiellauf durchzuführen und zu dokumentieren, der mit kleineren Zahlen arbeitet und so erst einmal die Funktionsweise des Modells bzw. seiner Implementation übersichtlich demonstriert.

Abbildung 2 zeigt einen möglichen Verlauf der Infektion, wenn von individuell verschiedenen Ansteckungsquellen und -zielen („Freunde“) und folgenden Werten ausgegangen wird:

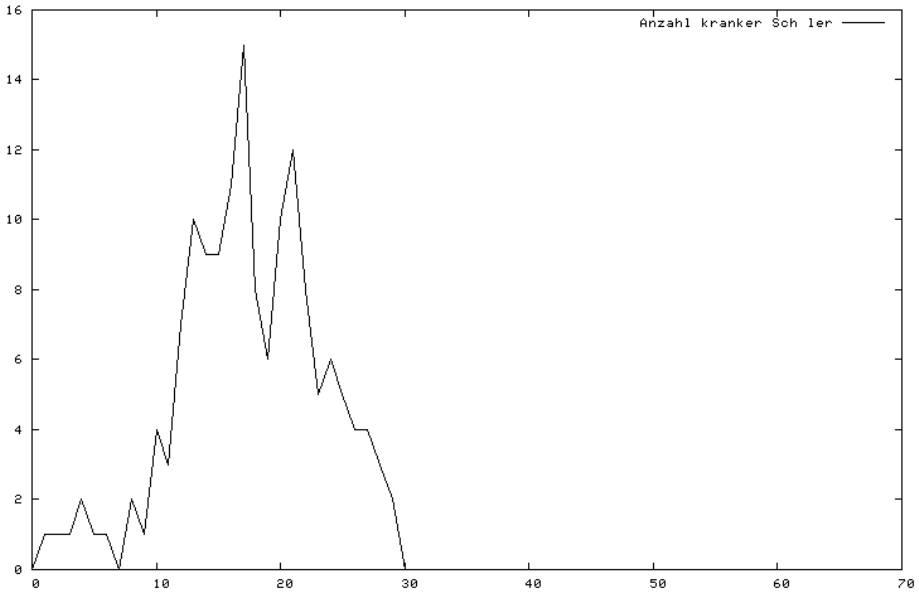


Abbildung 2: Spezieller Epidemierverlauf

```
# Eigenschaften der Krankheit:
# Sie dauert 9 Tage
dauer: 9
# Sie ist vom 3.-9. Tag ansteckend - ausgeschlossen sind
# aber die Tage, an denen man wirklich krank ist.
ansteckend: 3 7 8 9
# An den Tagen, an denen sie ansteckend ist, wird sie
# mit 30%iger Wahrscheinlichkeit an die Freunde übertragen.
wahrscheinlichkeit: 30
# Wirklich krank ist man am 4.-6. Tag.
krank: 4 5 6
```

Da der Zufall hier eine große Rolle spielt, ist es zumindest für die „Pflichtwerte“ sinnvoll, zahlreiche Läufe durchzuführen und einen mittleren Verlauf zu berechnen. Wenn man das Programm zu obigem Modell und mit den Pflichtwerten mehrere hundert Male aufruft und Mittelwerte über die einzelnen Tage bildet, sieht die Kurve aus wie in Abbildung 3. Diese Graphik sagt uns, dass die Schülerin mit gutem Gewissen innerhalb der ersten Woche ihre Party feiern und davon ausgehen kann, dass 35 von 38 Freundinnen gesund genug sind, um mitzufeiern. Ebenso ist nach 30-25 Tagen die Wahrscheinlichkeit recht hoch, dass sie ihr „Gästeziel“ erreichen kann.

Bewertungskriterien

Die Bewertung einer Lösung dieser Aufgabe orientiert sich unmittelbar an den recht präzise gestellten Teilaufgaben.

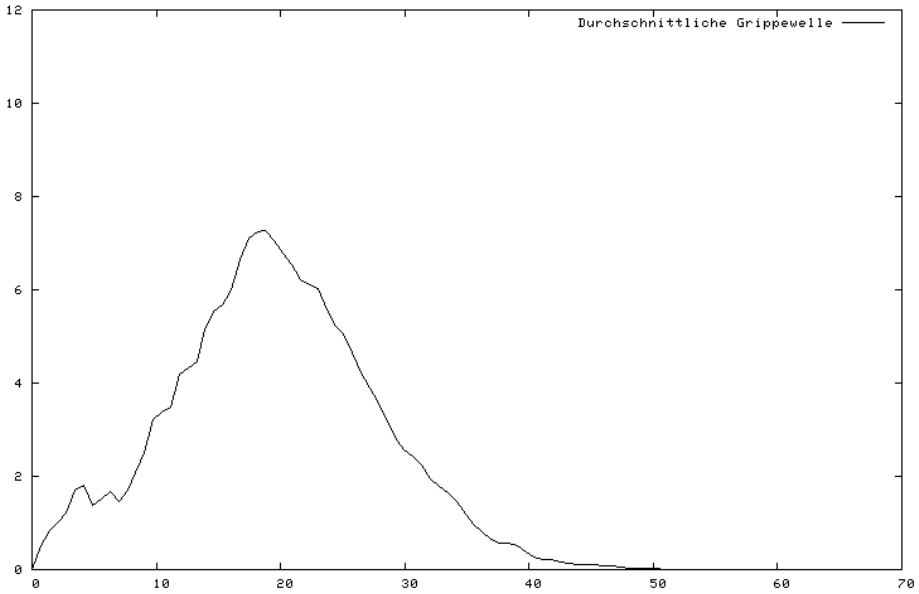


Abbildung 3: Durchschnittlicher Epidemieverlauf

- Teil 1: Die Entscheidungen bei der Wahl des Modellierungsansatzes müssen erkennbar werden und begründet sein. Es sollte darauf eingegangen werden, warum das Modell einer realen Grippewelle möglichst nahe kommt.
- Teil 1: Der Modellierungsansatz sollte zumindest den individuellen Krankheitsverlauf und eine Ansteckungsrate berücksichtigen.
- Teil 2: Die Auswahl des Implementationswerkzeugs muss kurz begründet sein. Ein Argument wie „Ich habe Programmiersprache XY gewählt, weil ich sie am besten beherrsche.“ ist nicht stichhaltig.
- Teil 2: Der Modellierungsumsatz muss korrekt implementiert sein.
- Teil 3: Die Tabellen und Grafiken stellen den Epidemieverlauf geeignet dar, es reichen aber (wie oben) auch Grafiken alleine.
- Teil 3: Das vorgegebene Beispiel muss, wie bei allen Aufgaben, natürlich bearbeitet sein. Bei Verwendung von Zufallswerten ist es außerdem wichtig, den Verlauf der Grippewelle mehrfach zu simulieren, um zu verlässlichen Ergebnissen zu kommen.
- Teil 4: Die Modellbewertung ist ein wichtiger Teil. Reichweite und Qualität des Modells sollten untersucht worden sein, wobei die Untersuchung Gründe für die Einschätzung dieser beiden Aspekte liefern sollte. Die Frage nach der Übertragbarkeit des Modells muss beantwortet sein.

Aus den Einsendungen: Perlen der Informatik

Allgemeines

Nun die Öffne-Funktion: `void Schließen ()`

Es wäre daher gut, wenn Sie das Program auch unter Win 2000 starten, da hier im DOS-Fenster ein Scroll-Balken ist. *Also* das *ist der entscheidende Fortschritt bei Windows 2000!*

Dämliches DOS! Das ist das letzte Mal, dass ich 'ne Konsolenanwendung geschrieben hab – warum geht denn da kein Copy & Paste?!

Anm.: Damit die werte Jury durch vermutlich viele sehr ähnliche Programmfragmente verschiedener Einsender nicht zu gelangweilt ist, habe ich auf unkonventionelle Weise im gesamten Programm die Spaltenindizes mit y und die Zeilenindizes mit x bezeichnet, in der Erwartung, dass eine waagrechte y- und senkrechte x-Achse dem ganzen Sourcecode einen revolutionär avantgardistischen Touch verleiht.

Um das Programmdesign noch schlechter aussehen zu lassen, habe ich dann noch fünf globale Variablen eingeführt.

So, ich hoffe, das ist jetzt verständlich geworden, besser kann ich's nicht erklären – nächstes Mal schicke ich ein interaktives Video mit.

In wieweit die zugehörige Dokumentation Sinn macht, weiss ich selbst nicht so genau.

Mein Programm besteht grob aus zwei Teilen. Nennen wir sie mal den nützlichen und den überflüssigen Teil (welcher jetzt was ist, überlass ich euch).

Auch bietet meine Schule (Bayern!!!) keinerlei Informatik-Unterricht, in dem man entsprechende Vorgehensweisen erlernen könnte

. . . da dies oft nicht der Fall ist, . . . kann man nicht die Mathematik benutzen. Daher ist es erst nötig, ein Programm zu schreiben.

. . . wie das Programm funktionuckelt.

Fucktion *Ja ja, nach zwei Monaten asketischer Bearbeitung der BWINF-Aufgaben denken manche eben nur noch an das eine.*

Aufgabe 1

Um das Problem des Dieners von Graf Rüdeger zu lösen, ist es am besten, man ändert die Zustände der Riegel nach den entsprechenden Regeln so lange, bis alle Riegel geschlossen sind. *Um das Ziel zu erreichen, gehe den Weg bis zum Ziel – OK, aber wie?*

Da nur die niedrigen Riegel interessieren, ist es sinnvoll, vom höchsten anzufangen und die anderen zu bewegen.

Aufgabe 2

Man könnte so einen Roboter bauen, der die Gäste dann zu ihren Zimmern führen oder ihre Koffer tragen kann. Praktisch wäre ebenso ein Scanner für den Fingerabdruck oder einfach eine Stimmkontrolle.

Die entlassenen Arbeitskräfte werden den Reservierungsrechner über alles in der Welt hassen.

Fehlehrekennunksystem *Am besten gleich auf sich selbst anzuwenden.*

. . . um dem Anrufer die gewünschte Antwort abzuringen.

(Das Programm) verwendet Bitshausen, da mir -hafen nicht gefällt und Perlenmeer eh nur ein Marketingname ist.

Dieser (Mitarbeiter) kann nichts anderes tun als zuhören und versuchen, den Anrufer zu beschwichtigen (professionelles Beschwerdemanagement).

Wenn ein Mensch die selbe Logik hätte und die selbe Sprache wie ein Datenbanksystem sprechen würde, könnte man auf das nun folgende Reservierungssystem komplett verzichten.

Aufgabe 3

Nach vielen Überlegungen stellt man schließlich fest, dass es für dieses Problem wahrscheinlich unmöglich ist, einen Algorithmus zu finden. Also muss auf den „Brute-Force“-Ansatz zurückgegriffen werden.

Als ein Ergebnis allein des 80maligen Generierens aller möglichen Wegstrings kam ich dann also auf eine Dauer von schlappen 238000 Trilliarden Jahren. „Lass dich nicht entmutigen, wenn dein Programm für dein Gefühl zu lange rechnen sollte.“ Haha!

Setzt man dafür ein paar Werte ein, ergeben sich exorbitant große Zahlen. Der Zerfall sämtlicher Materie im Universum in ihre subnukleotiden Bestandteile wäre abgeschlossen, bevor mein PC für eine Wabe, so groß wie das angegebene Beispiel, berechnet hätte, dass es keinen BWInF gibt.

Aufgabe 4

„If you´ve got a shiny new hammer every problem looks like a nail.“ Wenn Sie auch eines meiner zwei anderen Programme korrigiert haben, können Sie sicherlich schon erraten, wie meine Lösungsidee aussieht: Rekursion! *Juhu, Überraschung!*

Aufgabe 5

Wenn man die Immunität verlierende Rate auf 0 setzt, könnte man die Immunen auch als Gestorbene betrachten, wobei Immunwerdende von Genesenden dann die Sterberate angibt.

Hysterische Mütter, die in den Nachrichten oder Zeitungen etwas von einer drohenden Grippewelle hören, könnten ihre Kinder zum nächsten Arzt schicken, der sie gegen die aktuell auftretenden Grippen impft.

Krippewelle

Herzlich Willkommen zum Prognose-Programm für Krippen-Erkrankte.

Um eine Tabelle möglichst schnell in die Tat umzusetzen, sollte man, laut Lehrers Stimme, sie sowieso mit Excel gestalten, denn schneller + genauer + gutaussehender = obereffektiv.

Also sollte sie ihre Freier so schnell wie nur möglich feiern.

Dies geschieht aus dem Grund, da die Funktion Immunisierung phasenverschoben die Neuerkrankten heilt. Wenn diese (die Kranken) nun aber schon bei einem Wert beginnen, kann dann die Funktion sie nicht heilen, denn die Kranken wären dann nicht von Neuerkrankten erzeugt worden.

Leider werden die 4000-8000 Grippetoten (aus Mitgefühl) nicht in der Formel berücksichtigt. OK, 90% kommen aus der Risokogruppe der über 60-jährigen. Aber man kann ja nie wissen.

Am 2. Tag sind 48 der 38 Schüler krank. . . . Am 13. Tag sind -18 der 38 Schüler krank.