28. Bundeswettbewerb Informatik 2009/2010

Die Aufgaben der zweiten Runde

Allgemeine Hinweise

Herzlichen Glückwunsch zum Erreichen der zweiten Runde! Hier sind die Aufgaben. Sie sind anspruchsvoll, und ihre Bearbeitung ist aufwändig. Aber die Mühe lohnt sich, denn durch Teilnahme an der zweiten Runde

- wirst du sicher sehr viel lernen;
- kannst du dich für die Endrunde qualifizieren;
- wirst du als Endrundenteilnehmer vielleicht zu weiteren Veranstaltungen eingeladen;
- kannst du mit einer guten Leistung einen Buchpreis des Verlags O'Reilly gewinnen;
- hast du am Ende eine Arbeit fertig gestellt, die du als so genannte „Besondere Lernleistung" in die Abiturwertung einbringen kannst;
- kannst du dich (als jüngerer Teilnehmer) um die Teilnahme an einer Deutschen Schülerakademie bewerben;
- hast du, falls du aus der südlichen Hälfte Deutschlands stammst, die Chance auf die Teilnahme an einem Workshop beim Max-Planck-Institut für Informatik in Saarbrücken.

Wir wünschen also viel Spaß und viel Erfolg bei der Bearbeitung!

Wichtig: An dieser Runde dürfen nur Einzelpersonen teilnehmen, die in der ersten Runde in drei Aufgaben mindestens 12 Punkte erreicht oder einer Gruppe angehört haben, der dieses gelungen ist. Gruppenarbeit ist in der zweiten Runde nicht zulässig.

Einsendeschluss ist der 12. April 2010; es gilt das Datum des Poststempels. Bitte sende deine Lösungen wieder an den **Bundeswettbewerb Informatik, Ahrstraße 45, 53175 Bonn**.

Es gibt drei Aufgaben. **Wichtig**: Eine Einsendung darf nur Bearbeitungen zu höchstens zwei Aufgaben enthalten, deren Bewertung dann das Gesamtergebnis ausmacht. Sollte eine Einsendung Bearbeitungen zu allen drei Aufgaben enthalten, werden wir zwei davon zufällig auswählen und nur diese bewerten.

Die Bearbeitung einer Aufgabe sollte zunächst eine einfache, nachvollziehbare und vollständige Lösung aller Teilaufgaben enthalten. **Pluspunkte** für eine höhere Bewertung kannst du erreichen, wenn du die Aufgabe dort, wo es möglich und sinnvoll ist, eigenständig weiterentwickelst. Sinnvoll sind inhaltliche Erweiterungen und Verbesserungen, etwa von Datenstrukturen und Algorithmen; uninteressant sind aufwändige Tricks, z.B. zur reinen Verschönerung der Bedienungsoberfläche. Begründe für jede Erweiterung, weshalb sie sinnvoll ist und ihre Realisierung eine eigene Schwierigkeit darstellt.

Denke bitte daran, dass zur Bewertung möglicherweise nur die Papierunterlagen herangezogen werden können. Diese sollten also einen lückenlosen und nachvollziehbaren Nachweis

des Leistungsumfangs und der Funktionstüchtigkeit der Programme geben. Der Umfang der Einsendung soll sich in Grenzen halten; eine gute Dokumentation vermittelt kurz und präzise alles Nötige, insbesondere die wesentlichen Lösungsideen. Nötig ist alles, was Interessierte mit guten Informatikkenntnissen, die die Aufgabenstellung kennen, wissen müssen, um die Lösungsidee zu verstehen und die Realisierung dieser Idee nachzuvollziehen. Generell sind zwar gute und originelle Ideen entscheidend, aber die Dokumentation hat schon oft den Ausschlag für oder gegen das Weiterkommen gegeben.

Grundsätzlich gelten die Gliederungs- und Dokumentationsrichtlinien der 1. Runde weiter. Zu jeder Teilaufgabe gehört also die Lösungsidee und die Dokumentation der Lösung sowie des dazu gehörigen Programms (eine Beschreibung, wie die Idee in konkrete Programmelemente umgesetzt wurde, Hinweise auf Nutzungsgrenzen, Besonderheiten usw.). Dabei sind (halb-)formale Notationen besser als Programmausschnitte. Für die geforderten Programme erwarten wir Programmablaufprotokolle, also kommentierte Probeläufe des Programms, aus denen ersichtlich wird, wie das Programm sich in unterschiedlichen Situationen verhält. Sende uns außerdem bitte (abgedruckt!) aussagekräftige Ergebnisse von Programmläufen mit unterschiedlichen Daten, auch wenn Beispiele nicht explizit gefordert sind. Komplettiert wird das Papiermaterial durch den Programmtext, wobei unwichtige und automatisch generierte Teile nicht ausgedruckt werden sollen.

Schicke uns alles in lesbarer Form auf Papier, Schriftgröße mindestens 10 Punkt, bei Quelltext mindestens 8 Punkt. Verwende bitte lose, gelochte Blätter im Format DIN A4 (Hüllen mit Lochrand nur bei ausreichender Stabilität verwenden; keine Heftstreifen oder Mappen) und gib auf jedem Blatt Verwaltungsnummer, Vorname, Name und Seitennummer an. Die Verwaltungsnummer steht auf der Teilnahmebescheinigung der ersten Runde. Bitte gliedere deine Einsendung in (a) Allgemeines (zur Bearbeitung, Angaben zu technischen Voraussetzungen, persönliche Anmerkungen etc.), (b) Unterlagen zur ersten bearbeiteten Aufgabe und (c) Unterlagen zur zweiten bearbeiteten Aufgabe.

Außerdem sende uns bitte die Programmtexte und lauffähigen Programme auf einer CD oder DVD. Bei der Bewertung können Programme unter Windows (XP / Vista / 7), Linux und Mac OS X (10.6) ausgeführt werden.

Fragen zu den Aufgaben dürfen per E-Mail an `bwinf@bwinf.de` oder telefonisch unter 0228–378646 (zu üblichen Arbeitszeiten) gestellt werden. Die Antwort auf E-Mail-Anfragen kann sich leicht verzögern. Informationen zur 2. Runde finden sich auf den Webseiten des BWINF (`www.bwinf.de`). In der BWINF-Community auf `einstieg-informatik.de` wird sicher wieder über die Aufgaben diskutiert werden – ohne Lösungsideen auszutauschen.

Allen Teilnehmern der zweiten Runde wird Anfang Juni die Bewertung mitgeteilt. Die Besten werden zur Endrunde eingeladen, die vom 4. bis zum 8. Oktober 2010 am Institut für Informatik der Universität Freiburg ausgerichtet werden wird. Dort werden die Bundessieger und Preisträger ermittelt und am letzten Tag ausgezeichnet. Bundessieger werden in die Förderung der Studienstiftung des deutschen Volkes aufgenommen. Außerdem werden Geld- und Sachpreise vergeben. Der Rechtsweg ist ausgeschlossen.

Zum Schluss noch einmal: Viel Spaß und viel Erfolg!

MCI-Sonderpreis

Beim Bundeswettbewerb Informatik spielt die Mensch-Computer-Interaktion und damit die Benutzungsschnittstelle eines eingesandten Programms für die Bewertung prinzipiell keine Rolle. Für die Benutzbarkeit von Informatiksystemen ist diese Komponente aber von ganz entscheidender Bedeutung, und so wird bei einigen Einsendungen zum BWINF erhebliche Mühe auf den Interaktionsaspekt verwendet.

Diese Mühe soll belohnt werden: Der Fachbereich Mensch-Computer-Interaktion (MCI) der Gesellschaft für Informatik (GI) schreibt erneut einen Sonderpreis für besonders gelungene Benutzungsschnittstellen aus. Verliehen wird dieser Preis auf der Tagung „Mensch & Computer 2010“, die vom 12. bis zum 15. September 2010 in Duisburg stattfinden wird.

Für Bewerbungen um den MCI-Sonderpreis sollst du für eines deiner im Rahmen der Aufgabenbearbeitung entwickelten Programme eine besonders innovative und gebrauchstaugliche Benutzungsschnittstelle entwickeln – auch wenn die zugehörige Aufgabe die Entwicklung einer Benutzungsschnittstelle nicht zwingend erfordert. Die zu entwickelnde Benutzungsschnittstelle kann also zusätzliche, für die Softwarenutzung sinnvolle Funktionen beinhalten, die für die Lösung der eigentlichen Aufgabe nicht erforderlich sind.

Beschreibe die Benutzungsschnittstelle in einem separaten Dokument. Gib aber nicht nur eine Bedienungsanleitung, sondern erläutere vor allem Entwurfskonzept und -entscheidungen. Bewertet wird deine Bewerbung nach folgenden Kriterien:

Dialogkriterien: Der Dialog eines Benutzers (oder einer Benutzerin) mit einem System ist

aufgabenangemessen, wenn das System den Benutzer bei der Ausführung der Aufgaben unterstützt und sinnvoll führt, ohne Handlungen unnötig einzuschränken;

selbsterklärend, wenn jeder Dialogschritt durch Reaktionen des Systems sofort verständlich wird oder dem Benutzer auf Anforderung erklärt wird;

kontrollierbar, wenn der Benutzer in jeder Situation Richtung und Geschwindigkeit der Interaktion bestimmen kann, bis sein Ziel erreicht ist;

fehlertolerant, wenn trotz offensichtlicher Fehler in der Eingabe das beabsichtigte Ergebnis ohne bzw. mit nur geringem Eingreifen des Benutzers erzielt werden kann.

Präsentationskriterien: Die Präsentation von Information ist

fachlich gut gestaltet, wenn zusammengehörige Informationen räumlich gruppiert sind, ein „aufgeräumter“ Eindruck entsteht;

übersichtlich, wenn die Menge der Informationen knapp und strukturiert dargestellt wird;

lesbar, klar und präzise, wenn die (visuelle) Darstellung der Information leicht zu lesen bzw. erkennen ist und der Informationsgehalt ohne überflüssige Informationen vermittelt wird.

BWINF-Kriterien: Ein Programm und seine Bedienschnittstelle sind

originell, wenn die Bedienschnittstelle ungewöhnlich und mit eigenen Mitteln (aber dennoch ergonomisch) gestaltet ist;

inspizierbar, wenn die Bedienschnittstelle vollen Zugang zur Funktionalität erlaubt, also z. B. Kontrollflüsse dargestellt werden oder Parameter beeinflusst werden können.

Aufgabe 1: Universeller Öffnungscode

Ein selbst gebautes Sytem zum Authentisieren an einem technischen Gerät besteht aus der Sicht des Nutzers aus 25 Kippschaltern, die sich in der Stellung 0 oder 1 befinden können. Die Stellung aller Schalter lässt sich daher eindeutig durch einen 25-stelligen Bitstring darstellen, z.B. durch 0010110110110111001011111.

In Wirklichkeit werden nur fünf der 25 Schalter verwendet. Wenn sich diese fünf Schalter in der richtigen Stellung befinden, wird der Zugang gewährt. Diese Sparsamkeit hat zwei Gründe: Einerseits ist es billiger und einfacher, das Gerät mit nur fünf aktiven Schaltern zu bauen, und andererseits wäre es sehr lästig, sich die Stellungen von 25 Schaltern merken zu müssen und alle in die richtige Stellung zu bringen.

Eine Möglichkeit, ohne Kenntnis der aktiven Schalter und ihrer richtigen Stellung Zugang zu erreichen, ist das Ausprobieren *aller* möglichen Stellungen aller 25 Schalter. Wenn wir die Schalterstellungen durch Bitstrings darstellen, entspricht dies dem Durchprobieren aller $2^{25} = 33\,554\,432$ Bitstrings der Länge 25, welches doch recht viele sind.

Tatsächlich muss man nicht alle Kombinationen ausprobieren. Es genügen deutlich weniger Bitstrings, um sicher Zugang zu erhalten. Ein Beispiel: Bei neun Schaltern, von denen drei aktiv sind, genügen statt $2^9 = 512$ Bitstrings die rechts abgebildeten 16. Bei 18 Schaltern, von denen 4 aktiv sind, genügen statt 2^{18} (262 144) Bitstrings deutlich weniger als 100.

```
000000010
000101001
001011011
001101110
001110000
010011100
010110111
011101010
100010101
101001000
101100011
110001111
110010001
110100100
111010110
111111101
```

Eine Menge von Bitstrings der Länge N bezeichnet man als *universellen Öffnungscode* (für N Schalter, von denen M aktiv sind), wenn zu jeder möglichen Position der M aktiven Schalter und zu jeder Stellung dieser M Schalter die Menge mindestens einen Bitstring enthält, dessen Schalterstellungen damit übereinstimmen. Mit anderen Worten: Wenn wir verschiedene $i_1, i_2, \ldots, i_M \in \{1, \ldots, N\}$ und $p_1 \ldots, p_M \in \{0, 1\}$ wählen, dann muss ein universeller Öffnungscode einen Bitstring $b_1 b_2 \ldots b_N$ enthalten mit der Eigenschaft $b_{i_1} = p_1, \ldots, b_{i_M} = p_M$.

Aufgabe

Schreibe ein Programm, das für das oben beschriebene System mit 25 Schaltern und 5 aktiven Schaltern ($N = 25$ und $M = 5$) einen universellen Öffnungscode berechnet und diesen ausgibt. Versuche einen möglichst kurzen Öffnungscode zu erzeugen.

Aufgabe 2: Das Turmrestaurant

Das Turmrestaurant hat einen einzigen Tisch, dessen Fläche durch zwei konzentrische Kreise begrenzt wird. Die N Sitzplätze sind alle auf der Innenseite des Tisches angeordnet, so dass sämtliche Gäste des Restaurants beim Essen aus den Fenstern schauen können. Betritt eine Gruppe von k Gästen das Restaurant, muss der Ober k aufeinander folgende Sitzplätze am Tisch für sie finden; wenn die Gruppe gegessen hat, werden alle k Plätze gleichzeitig wieder frei. Falls keine k aufeinander folgenden freien Plätze für die Gruppe gefunden werden können, geht die Gruppe, ohne im Restaurant zu essen, und der Ober ärgert sich.

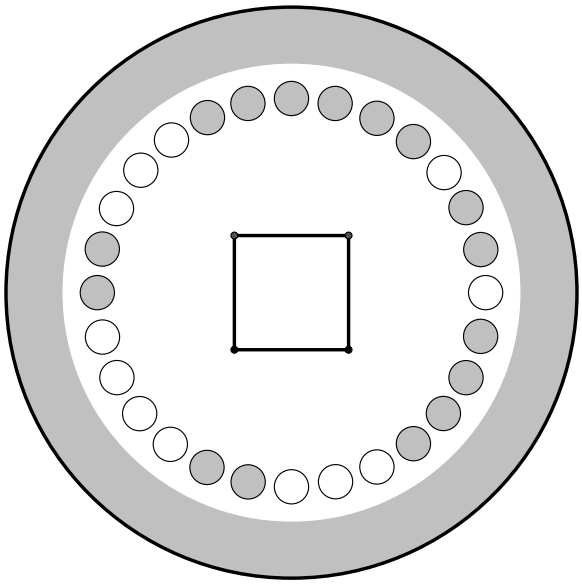

Abbildung 1: Das Turmrestaurant, hier mit $N = 28$ Plätzen. Zur Zeit gibt es maximal 4 aufeinander folgende freie Plätze.

Aufgabe

1. Schreibe ein Programm, das die Platzierung neuer Gruppen von Gästen im Restaurant übernehmen kann, und zwar so, dass der Ober sich möglichst selten ärgert.
2. Die Töchter des Obers haben einen Streich mit ihm vor, und die meisten ihrer Klassenkameraden machen mit. Die Schüler schließen sich zu Gruppen zusammen und essen im Turmrestaurant. Dabei versuchen sie, die Gruppen so zu bilden und wieder aufzulösen, dass das Restaurant bereits bei möglichst geringer Auslastung die nächste Gruppe nicht aufnehmen kann, weil die wenigen Gäste verstreut sitzen und keine hinreichend große Lücke zwischen ihnen vorhanden ist. Schreibe ein Programm, das den Schülern hilft, ihren Streich durchzuführen.
3. Lasse deine Programme aus den Teilen 1 und 2 „gegeneinander antreten“ und berichte, was passiert. Wie viele Schüler müssen teilnehmen, bevor sich der Ober überhaupt jemals ärgert?

Aufgabe 3: Anagramme

Es ist ein netter Zeitvertreib, die Buchstaben eines gegebenen Wortes so umzuordnen, dass ein anderes Wort entsteht – ein Anagramm. Zum Beispiel lässt sich für das Wort LEIB das Anagramm BEIL finden.

Schwieriger wird es, wenn Eingabe und Ausgabe aus mehreren Wörtern bestehen können. Dabei dürfen Leerzeichen in der Eingabe ignoriert und in die Ausgabe beliebig eingefügt werden. Ein „gutes“ Anagramm sollte dann eine Wortfolge sein, die auch in einem realen Text vorkommen könnte. Nur selten gelingt es, derart sinnvolle Anagramme zu finden.

Aufgabe

Schreibe ein Programm, das aus einer kurzen Eingabe eine überschaubare Menge von Anagrammen generiert. Das Ziel sollte sein, dass zumindest hin und wieder ein Anagramm darunter ist, das (mit gutem Willen) als sinnvoll betrachtet werden kann und vielleicht sogar lustig ist.

Ein unverzichtbarer Ausgangspunkt sind ein oder mehrere hinreichend umfangreiche Texte, denen du die möglichen Wörter und eventuell auch weitere für das Bilden sinnvoller Anagramme wichtige Informationen entnehmen kannst.

Du darfst dich auf englischsprachige Texte beschränken; in der englischen Sprache lassen sich sinnvolle Anagramme leichter finden als in vielen anderen Sprachen.

Beispiel: Aus dem Wort COMBINATORICS lässt sich das Anagramm TOMB IN CORSICA, aber auch MAC IN ROBOTICS generieren.