

Klausur Programmieren 1

HAW-Hamburg, Fakultät Technik und Informatik, Department Informations- und Elektrotechnik

Prof. Dr. Robert Heß, 12.7.2022, Bearbeitungsdauer: 180 Min.

Erlaubte Hilfsmittel: Vorlesungsunterlagen, Lösungen aus dem Praktikum und C/C++ Einführungsbücher.

Ergebnis: von 100 Punkten

Note: Punkte.

1 Einleitung

In einer Demokratie finden in regelmäßigen Abständen Wahlen statt, bei denen anschließend die Anzahl der erhaltenen Stimmen auf die Sitze im Parlament/Senat/Gemeinderat etc. umgerechnet werden. Dabei kommen je nach Wahlgesetz verschiedene Verfahren zum Einsatz.

In dieser Aufgabe soll das *Divisorverfahren mit Standardrundung* implementiert werden, auch benannt nach *Sainte-Laguë*. Dies kann wie folgt realisiert werden:

In einer Tabelle werden in der ersten Zeile die Stimmen aller Parteien eingetragen. In den Zeilen 2, 3, 4, 5 ... werden die Stimmen der ersten Zeile durch 3, 5, 7, 9 ... dividiert. Danach werden die n_{Sitze} größten Zahlen ermittelt, wobei n_{Sitze} für die Anzahl der zu vergebenen Sitze steht. Jede gefundene Zahl bedeutet ein Sitz für die entsprechende Partei.

Als Beispiel diene die Wahl des Gemeinderats in Mallersdorf-Pfaffenberg im Landkreis Straubing-Bogen in Niederbayern im Jahr 2020. Die 67 173 gültig abgegebenen Stimmen verteilten sich auf fünf Parteien gemäß der ersten Zeile der gezeigten Tabelle. In der Tabelle ergibt sich eine Schwelle bei 1 659,18, ab der ein Sitz zugewiesen wurde. Alle Zahlen oberhalb dieses Werts sind in der Tabelle fett markiert, was einem zugewiesenen Sitz entspricht. In der letzten Zeile sind die sich ergebenden Summen der Sitze für die Parteien angegeben.

Teiler	Partei 1	Partei 2	Partei 3	Partei 4	Partei 5
1	28 206,000	18 251,000	10 000,000	9 229,000	1 487,000
3	9 402,000	6 083,667	3 333,333	3 076,333	495,667
5	5 641,200	3 650,200	2 000,000	1 845,800	297,400
7	4 029,429	2 607,286	1 428,571	1 318,429	212,429
9	3 134,000	2 027,889	1 111,111	1 025,444	165,222
11	2 564,182	1 659,182	909,091	839,000	135,182
13	2 169,692	1 403,923	769,231	709,923	114,385
15	1 880,400	1 216,733	666,667	615,267	99,133
17	1 659,176	1 073,588	588,235	542,882	87,471
Sitze:	8	6	3	3	0

In dieser Klausur soll ein Programm erstellt werden, welches zunächst die abgegebenen Stimmen vom Benutzer abfragt und dann die Anzahl der Sitze für die Parteien ermittelt. Der Einfachheit halber wollen wir uns auf max. 10 Parteien und max. 50 Sitze beschränken. Auch wenn es nicht der einfachste Ansatz ist, soll hier mit einem zweidimensionalen Vektor gearbeitet werden. Auf den Sonderfall, dass sich für die Sitze n_{Sitze} und $n_{\text{Sitze}} + 1$ die gleiche Zahl ergibt, gehen wir in dieser Aufgabe nicht weiter ein.

2 Programmieraufgaben

Aufgabe 1 (5 Punkte)

Erstellen Sie zwei Funktionen zur sicheren Benutzerabfrage von ganzen Zahlen. Die eine prüft auf Ober- und Untergrenze, die andere lässt den Wertebereich unberücksichtigt.

Aufgabe 2 (10 Punkte)

Fragen Sie im Hauptprogramm die Anzahl der Parteien (2-10) und der Sitze (10-50) ab. Danach soll das Programm die Anzahl der Stimmen für alle Parteien abfragen und in einem Vektor speichern, wobei es Ihnen überlassen ist, ob Sie dies in einer Funktion implementieren.

Aufgabe 3 (25 Punkte)

Erstellen Sie eine Funktion, die anhand der Stimmenverteilung die Anzahl der Sitze für alle Parteien berechnet. Der Funktion soll drei Parameter erwarten: a) Vektor mit der Stimmenverteilung, b) Anzahl der Sitze n_{Sitze} und c) Vektor zum Speichern der Sitzverteilung.

Legen Sie innerhalb der Funktion einen 2d-Vektor für die maximale Anzahl an Parteien (10) und Sitze (50) an und speichern Sie darin die Tabelle gemäß der Einleitung. Danach ermitteln Sie die n_{Sitze} größten Zahlen und speichern die Anzahl der Sitze in den entsprechenden Parameter.

Tipp: Suchen Sie n_{Sitze} mal die Tabelle nach der größten Zahl durch. Um nicht die selbe Zahl immer wieder zu finden, setzen Sie das Tabellenelement auf null, und erhöhen für die entsprechende Partei die Anzahl der Sitze um eins.

Aufgabe 4 (20 Punkte)

Erstellen Sie eine Funktion, welche die Sitzverteilung ordentlich auf dem Bildschirm ausgibt.

Die Tabelle soll vier Spalten beinhalten:

Spalte 1: Nummer der Partei

Spalte 2: Anzahl der Stimmen

Spalte 3: Anzahl der Sitze nach der beschriebenen Methode

Spalte 4: Anteilige Anzahl der Sitze auf drei Stellen nach dem Komma: Anzahl der Stimmen geteilt durch die Summe aller Stimmen mal die Anzahl der Sitze.

Aufgabe 5 (15 Punkte)

Fügen Sie die verschiedenen Programmstücke zu einem lauffähigen Programm zusammen. Achten Sie auf eine gute Benutzerführung und geben Sie eine Überschrift und Schlussmeldung aus.

Für alle Teilaufgaben gilt: Achten Sie auf einen guten Programmierstil (Vermeidung globaler Variablen, sinnvolle Variablennamen, Quellcode einrücken und kommentieren, keine absoluten Sprünge mit `goto`, keine Warnungen vom Compiler etc.).

3 Verständnisfragen

Aufgabe 6 (3 Punkte)

Nennen Sie drei Beispiele für unäre Operatoren.

Nachname: Vorname: MatrNr.:

Aufgabe 7 (8 Punkte)

Rechnen Sie die Zahl 23_{10} ins duale Zahlensystem um. Zeigen Sie Ihren Lösungsweg und verwenden Sie die Methode aus unserer Vorlesung.

Aufgabe 8 (4 Punkte)

Sie wollen alle Elemente eines Vektors mit einer Schleife auf null setzen. Welche ist zu bevorzugen? Begründen Sie Ihre Antwort.

- for** ... **while** ... **do** ...

Aufgabe 9 (4 Punkte)

Für die Werte einer ganzen Zahl soll zwischen vier Fällen unterschieden werden: a) der Wert null, b) der Wertebereich 10–20, c) der Wertebereich 50–100 und d) alle anderen Werte. Welche Art der Verzweigung ist zu bevorzugen? Begründen Sie Ihre Antwort.

- einzelne **ifs** **else if** Verkettung **switch** Verzweigung

Aufgabe 10 (3 Punkte)

Warum kann am Ende einer void-Funktion das `return` weggelassen werden?

Aufgabe 11 (3 Punkte)

Definieren und füllen Sie einen Vektor für die Fakultäten von 0 bis 6.