Technische Universität München
Institut für Informatik
Prof. Dr. Hans-Joachim Bungartz
Hayden Liu Weng
Sebastian Wolf
Michael Obersteiner

WiSe 2021/22

Numerisches Programmieren, Übungen

1. Übungsblatt: Zahlendarstellung, Rundungsfehler

1) Umrechnung von Zahlen

a) Schreiben Sie die ganzen Zahlen 19, 47 und 511 in binärer und hexadezimaler Darstellung!

b) Schreiben Sie die Brüche $-\frac{3}{8}$ und $\frac{1}{10}$ als Binärzahl!

2) Assoziativgesetz

Eine Eigenschaft, die bei Gleitkommazahlen leider verloren wird, ist das Assoziativgesetz. Betrachten Sie eine binäre Gleitkomma-Darstellung mit 4 signifikanten Stellen.

Beispiele:

Zahl	Anzahl signifikanter Stellen
1000000.0_2	8
$1 \cdot 2^6$	1
1000.1_2	5
-0.01011_2	4
10.10_2	4

Berechnen Sie im gegebenen Format die Werte

- $(-8 + 11) + 0.75$ und
- $-8 + (11 + 0.75)$.

Runden Sie dabei nach jedem Rechenschritt. Was ist zu beobachten?

3) Gleitkomma-Zahlen

In dieser Aufgabe soll Schritt für Schritt eine reelle Zahl in eine Maschinenzahl umgewandelt werden. Die umzuwandelnde Zahl lautet $-\frac{11}{10}$.

a) Schreiben Sie die Zahlen zuerst in eine andere, standardisierte Form folgender Gestalt um! An erster Stelle steht das Vorzeichen. Dann folgt der Betrag der Zahl in binärer Exponentialdarstellung, beginnend mit »$1, \ldots$«. Es geht weiter mit den Nachkommastellen, einem Malpunkt und abschließend steht eine Zweierpotenz (Beispiel: $-1,11001 \cdot 2^{-56}$).

b) Wandeln Sie die Ergebnisse von Teilaufgabe a) in Binärcode um! Dazu verwenden Sie den im Folgenden spezifizierten 32-Bit IEEE-Standard:

- Das erste Bit bestimmt das Vorzeichen: Eine Null bedeutet positive Zahl, eine Eins

bedeutet negative Zahl.

- Die nächsten 8 Bits sind für den Exponenten reserviert. Die gespeicherte Binärzahl entspricht dem Exponenten plus 127. Die Bit-Kombinationen 00000000 (entspräche einem Exponenten von -127) und 11111111 (entspräche $+128$) sind allerdings für spezielle Werte reserviert (0, ∞, NaN, ...).
- Die letzten 23 Bits dienen der Speicherung der Nachkommastellen. Dabei wird wie in Teilaufgabe a) von einer Normalisierung mit führender Eins ausgegangen (die nicht gespeichert werden muss). Genügen 23 Bits Genauigkeit für die Mantisse nicht, so wird zum nächstgelegenen darstellbaren Wert gerundet.
- Es gäbe noch viel über IEEE zu sagen, was aber nicht zur Lösung dieser Aufgabe notwendig ist. Weitere Informationen finden Interessierte zum Beispiel im Artikel »What Every Computer Scientist Should Know about Floating-Point Arithmetic« von David Goldberg.

c) Wir haben nun eine Zahl gesehen, welche mit dem in Teilaufgabe b) beschriebenem IEEE-Standard nicht exakt darstellbar ist. Wir wollen nun den Fehler einer solchen Rundung berechnen. Zur Vereinfachung der Rechnung werden wir uns die Zahl $x = 1 + 2^{-30}$ anschauen. Wie sieht die gerundete Zahl aus? Berechnen sie außerdem den absoluten und relativen Fehler.

d) Wie groß ist die Maschinengenauigkeit ε_{Ma} für den in Teilaufgabe b) beschriebenen IEEE-Standard? Durch welche Größe wird die Maschinengenauigkeit verändert? Welche andere Größe gibt es und was wird hierdurch reguliert?

e) Wie unterscheidet sich der Zahlenbereich eines IEEE **float** und eines **signed int**? Geben Sie hierfür den Zahlenbereich der Gleitkommazahlen getrennt für positive und negative Zahlen an! Geben Sie außerdem an wie sich die Schrittweite der Diskretisierung verhält.

f) Geben sie die erste Ganzzahl an die mithilfe des 32-Bit IEEE-Standards für Gleitkommazahlen nichtmehr exakt darstellbar ist. Welche Probleme entstehen hieraus für Programmiersprachen, welche nur mit Gleitkommazahlen arbeiten und keine Integerzahlen besitzen (z.B. Javascript).

4) Zusatzaufgabe: Ermittlung von π nach Archimedes

Viele mathematische und naturwissenschaftliche Probleme können nicht oder nur schwer durch Angabe einer direkten Lösungsformel gelöst werden, stattdessen ist aber oftmals eine schrittweise Annäherung an die exakte Lösung möglich. Solche sogenannten iterativen Approximationen lassen sich meist algorithmisch beschreiben und in ein Computer-Programm umsetzen. Dabei ist allerdings große Sorgfalt geboten, wie die folgende Aufgabe zeigt.

Ein klassisches Beispiel für die iterative Approximation ist die Bestimmung der Kreiszahl π. Eines der ersten bekannten Verfahren geht auf Archimedes von Syrakus (um 250 v.Chr.) zurück. Die Formel Kreisumfang gleich zweimal Kreisradius mal π war Archimedes bereits bekannt. Durch immer genauere Approximation des Umfangs eines Kreises mit Radius eins mit Hilfe von in den Kreis einbeschriebenen Polygonen konnte er somit π approximieren. Für ihn war das eine mühselige Arbeit mit Tinte und Papyrus, wir können das heute dank Computer im Bruchteil einer Sekunde erledigen.

a) Berechnen Sie Seitenlänge und Umfang eines in den Einheitskreis einbeschriebenen Quadrats (vgl. Abbildung 1)!

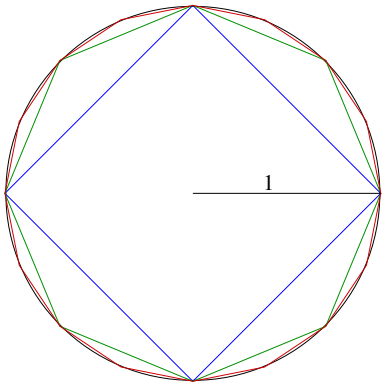

Abbildung 1: Einheitskreis und einbeschriebene Polygone

b) Durch Verdopplung der Ecken erhält man aus dem Quadrat sukzessive regelmäßige Achtecke, Sechszehnecke, etc. (vgl. Abbildung **??**). Geben Sie eine Formel für die Seitenlänge s_{n+1} des 2^{n+1}-Ecks in Abhängigkeit von der Seitenlänge s_n des 2^n-Ecks an!

Hinweis: Fertigen Sie eine Skizze an und erinnern Sie sich an den Satz von Pythagoras!

c) Mit der in Teilaufgabe b) gefundenen iterativen Formel für die Seitenlänge der Polygone, kann man nun schrittweise Näherungen für π berechnen. Implementiert man die Formel zum Beispiel in einer `FOR`-Schleife, so ergeben sich aber folgende Werte:

Anzahl der Ecken	2^5	2^{10}	2^{20}	2^{25}	2^{30}
Approximation	$3,136548$	$3,141587$	$3,141596$	$3,142451$	$0,000000$

Welcher Teil der Formel aus Teilaufgabe b) ist für den auftretenden Fehler verantwortlich? Beheben Sie das Problem durch algebraische Umformungen!

5) Zusatzaufgabe: Fehlerabschätzung von Zeitschrittweiten

In Computerspielen werden für die Simulation von Physik oft Zeitschrittweiten Δt von $\frac{1}{50}$ oder $\frac{1}{60}$ gewählt um die Positionsänderung von einem Objekt innerhalb eines Frames zu simulieren.

Wir wollen hier den vereinfachten Fall mit konstanter Objekt-Geschwindigkeit und unter Vernachlässigung aller externen Kräfte und Kollisionen betrachten. Dann lässt sich die Positionsänderung eines Objektes innerhalb eines Zeitschrittes durch $\Delta x = \Delta t \cdot \vec{v}$ beschreiben, wobei Δx die Positionsänderung innerhalb eines Zeitschrittes ist und $\vec{v}$ die konstante Geschwindigkeit des Objektes.

Durch die vorangegangenen Aufgaben haben wir bereits festgestellt, dass Zahlen in Binärdarstellung nur eine begrenzte Genauigkeit besitzen, womit Δt evtl. nicht exakt gespeichert werden kann.

Geben Sie eine Abschätzung an, ob und wieviel Genauigkeit bei der Konvertierung der Zeitschrittweite $\frac{1}{60}$ zu 32-bit IEEE floating-point Zahlen verloren geht.

Inwiefern kann sich dieser Fehler auf sehr lange Simulationsläufe auswirken? Haben Sie eine Idee, dieses Problem zu umgehen bzw. den entstehenden Fehler zu klein zu halten?

6) Zusatzaufgabe: Ganzzahlen, Fest- und Gleitkommazahlen

In dieser Aufgabe schauen wir uns die Vorteile von Gleitkommazahlen gegenüber Festkommazahlen und den IEEE-Standard an. Binär Gleitkommazahlen haben das folgende format:

$$(-1)^{\text{Vorzeichen}} \cdot 2^{\text{Exponent}} \cdot \text{Mantisse}.$$

Wir vergleichen drei Arten in denen wir Zahlen mit 8 Bits speichern können:

- I: **Ganzzahlen** – 1 Bit für Vorzeichen (0 für »+«, 1 für »-«) und 7 Bits für den Ganzteil (in dieser Reihenfolge),
- F: **Festkommazahlen** – 1 Bit für Vorzeichen, 4 Bits für den Ganzteil und 3 Bits für die Nachkommastellen (in dieser Reihenfolge),
- G: **Gleitkommazahlen** – 1 Bit für Vorzeichen, 5 Bits für den Exponenten und 2 Bits für die Mantisse (in dieser Reihenfolge).
  In diese Aufgabe betrachten wir den Exponent als vorzeichenbehaftete Ganzzahl.
  Bei der Mantisse wird von einer Normalisierung mit führender Eins ausgegangen, die nicht gespeichert wird.

Aufgabe: Füllen Sie die fehlende Einträge der Tabelle aus:

Nr.	Frage		$A = I$	$A = F$	$A = G$
1	Darstellungs-beispiele	1 0000000	-0	$-0,0$	$-1 = -2^{+0} \cdot 2^{0}$
2		0 0000000	$+0$		
3		0 1000000	64		
4		0 1000100	68		
5		0 1000110	70	$8,75$	
6		0 1000111	71	$8,875$	$0,875 = 2^{-1} \cdot (2^{0} + 2^{-1} + 2^{-2})$

In Aufgaben Nr. 22–29 wird die Rundungsfunktion $\text{rd}_\text{A} : \mathbb{R} \to A$ gebraucht. Für jedes $A \subset \mathbb{R}$,

ist sie definiert wie folgt:

$$\mathrm{rd}_{\mathrm{A}}(x) = a \in A, \text{ so dass } |x-a| \leq |x-b| \quad \forall b \in A \quad . \tag{1}$$

Zum Beispiel $\mathrm{rd}_{\mathrm{F}}(\pi) = 3,125$.

Nr.	Frage		$A = I$	$A = F$	$A = G$
7	Allg. Eigenschaften	$\lvert A\rvert$	255		
8		$\max_{a\in A} a$	127		
9		$\min_{a\in A} a$	-127		
10		$\min_{a\in A, a>0} a$	1		
11		Ist 0 in A?	ja		
12	Anzahl Zahlen von A in	$[2^{-15}, 2^{-14})$	0		
13		$[1, 2)$	1		
14		$[2, 4)$	2		
15		$[4, 8)$	4		
16		$[8, 16)$	8		
17		$[16, 32)$	16		
18		$[32, 64)$	32		
19		$[64, 128)$	64		
20		$[2^{15}, 2^{16})$	0		
21		$[0, 1)$	1		
22	Rundungen: $\mathrm{rd_A}(x)$	3^{-5}	0		2^{-8}
23		2,1	2		
24		3,1	3		
25		9	9		
26		18	18	15,875	
27		1023	127	15,875	
28	Absoluter Rundungsfehler: $\lvert x - \mathrm{rd_A}(x)\rvert$	3^{-5}	3^{-5}		
29		2,1	0,1		
30		3,1	0,1		
31		9	0		
32		18	0		
33		1023	...	...	
34	Relativer Rundungsfehler: $\left\lvert\frac{x-\mathrm{rd_A}(x)}{x}\right\rvert$	3^{-5}	1		
35		2,1	0,048		
36		3,1	0,032		
37		9	0		
38		18	0		
39		1023	...	...	

Beobachtungen

a) Betrachten Sie Ihre Antworten zu Fragen Nr. 2 und 3 für $A = G$. Was ist zu beobachten?

b) Betrachten Sie Ihre Antworten zu Fragen Nr. 25 und 26 für $A = G$. Was ist zu beobachten?

c) Betrachten Sie Ihre Antwort zu Frage Nr. 25 für $A = G$. In manche Programmiersprachen (z.B. JavaScript) gibt es kein Format für Ganzzahlen. Welche Konsequenzen kann das haben?

d) Was ist die kleinste Ganzzahl, die bei 32-Bit IEEE Gleitkommazahlen nicht exakt darstellbar ist?

e) Betrachten Sie die Antworten zu den Fragen 26.-27. für $A = F$. Welche Konsequenzen kann die Abwesenheit einer Inf-Darstellung haben?

f) Für eine Mantissendarstellung mit 2 Bits ist die Maschinengenauigkeit $\varepsilon_{\mathrm{Ma}} = 2^{-3}$. Vergleichen Sie ihre Antworten zu Fragen Nr. 34-39 mit der Maschinengenauigkeit. Ist

$$\left| \frac{x - \mathrm{rd}_{\mathrm{G}}(x)}{x} \right| \leq \varepsilon_{\mathrm{Ma}} \tag{2}$$

immer erfüllt?

g) Gibt es eine Relation zwischen der Maschinengenauigkeit und der Zahl aus Frage Nr. 10?

h) Was ist der Anteil an Zahlen im $[0, 1)$ beim Format G (ungefähr)? Bei 32-Bit IEEE?

i) Wie viele Zahlendarstellungen sind für spezielle Werte (Exponentenkombinationen $00\ldots0$ und $11\ldots1$) bei IEEE reserviert?

j) Gibt es einen Unterschied zwischen I und F? Betrachten Sie Ihre Antworten zu Fragen 12-21 für $A = I$ und $A = F$. Kann man F anhand I implementieren?

k) Die darstellbare Zahlen bei I und F sind **linear** verteilt. Wie sind die Zahlen G verteilt?

l) Was hängt von der Anzahl an Bits in der Mantisse ab?

m) Was hängt von der Anzahl an Bits in dem Exponent ab?

Das einfache Format G ist leider an mehrere Stellen mehrdeutig. Alle Mehrdeutigkeiten müssen in einem Standard wie dem IEEE-Standard gelöst werden um Konsistenz und Reproduzierbarkeit zu garantieren.