PRAKTIKUM INFORMATIK I / PROJEKTAUFGABE 2B

Wichtiger Hinweis

Diese Aufgabe muss von den Gruppen bearbeitet werden, die in der Kalenderwoche 48 (ab Montag 25.11.) ihren Präsenztermin 2 haben.

Berechnung konvergierender Reihen

1.1 Einführung

In einem Computer-Prozessor müssen alle mathematischen Funktionen ausschließlich durch die Verwendung von Grundrechenarten berechnet werden. Die Grundrechenarten sind in Hardware realisiert. Funktionen wie Sinus, Cosinus oder die Exponentialfunktion können durch eine Reihenentwicklung ausgedrückt werden und damit durch Verwendung lediglich der elementaren Operationen Addition, Subtraktion, Multiplikation und Division approximiert werden.

Die Reihen gehorchen einem Bildungsgesetz, das als Summen-Formel definiert ist, wobei der *Laufindex* gegen Unendlich läuft. Die Reihen konvergieren gegen den exakten Wert der jeweiligen Funktion. Je nach dem wie viele Summanden bei der Berechnung tatsächlich verwendet werden (bei welchem Wert des Laufindex die Reihe abgebrochen wird), ist das Ergebnis eine mehr oder weniger genaue Approximation.

Der natürliche Logarithmus, zum Beispiel, kann als Potenzreihe mit Funktionsargument (1+x) und Laufindex k dargestellt werden:

$$\ln(1+x) = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{x^k}{k} = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \cdots$$

Diese ist allerdings nur für $|x| \leq 1$ gültig (|x| bezeichnet den Absolutbetrag des Wertes von x). Wenn x nicht in diesem Wertebereich liegt, soll mit Hilfe der Beziehung $ln(1+x) = -ln\left(1 + \left(\frac{-x}{1+x}\right)\right)$ eine Umrechnung erfolgen. Ist der x-Wert größer als 1 wird zunächst der Wert y = - x / (1+x) berechnet. Der Betrag von y ist in diesem Fall immer kleiner als 1. Deshalb kann ln(1+y) durch die vorstehend angegebene Reihe approximiert werden. Für die anschließende Berechnung von ln(1+x) gilt:

ln(1+x) = –ln(1+y).

1.2 Aufgabenstellung

Entwickeln Sie ein C Programm zur Berechnung des natürlichen Logarithmus. Dieses Programm soll eine bestimmte Anzahl positiver Gleitkommawerte vom Benutzer einlesen, für jeden dieser eingegebenen Werte den natürlichen Logarithmus näherungsweise berechnen und

das Ergebnis ausgeben. Als erste Eingabe, vor der Eingabe der Gleitkommawerte, erwartet das Programm eine ganze Zahl. Mit dieser Zahl wird die Anzahl der nachfolgend einzugebenden Gleitkommawerte festgelegt. Die Anzahl der einzugebenden Gleitkommawerte ist somit variabel.

1.3 Programmanforderungen

Beachten Sie bitte folgende Anforderungen an das Programm:

1. Die erste Benutzereingabe soll eine ganze Zahl sein, mit der die Anzahl der nachfolgend einzugebenden Gleitkommawerte festgelegt wird.
2. Nachdem das Programm die in Punkt 1 festgelegte Anzahl von Gleitkommawerten eingelesen und verarbeitet hat, soll es sich beenden.
3. Wenn vom Benutzer statt der geforderten Gleitkommawerte irgendwelche anderen Eingaben gemacht werden, z.B. Buchstaben statt einem Gleitkommawert, soll eine Fehlermeldung ausgegeben werden und das Programm soll sich beenden.
   Hinweis.: Die Funktion scanf liefert die Anzahl der korrekt eingelesenen Eingabeteile zurück, z.B. 1 wenn der scanf-Aufruf genau eine Zahl einlesen sollte und dies getan hat, siehe https://stackoverflow.com/questions/10469643/what-does-the-scanf-function-return - 10469649
   oder http://pronix.linuxdelta.de/C/standard_C/c_programmierung_6.shtml#4 .
   Wenn kein passender Wert für ein Formatelement eingelesen werden konnte, verändert scanf die entsprechende Zielvariable nicht. Auch damit kann man feststellen, ob eine korrekte Eingabe erfolgt ist.
   Die Fehlermeldung können Sie frei gestalten, es muss aber **genau eine Textzeile sein**, mit einem Ausrufezeichen (!) beginnen und den Grund für die vorzeitige Beendigung des Programms beschreiben.
4. Für jeden eingelesenen Wert muss das Ergebnis in folgender Form ausgegeben werden (genau eine Zeile für jede Eingabe): `ln(<Eingabe>) = <Ergebnis>, Iterationen: <Anzahl>`, wobei <Eingabe> durch den eingegebenen Wert und <Ergebnis> durch das berechnete Ergebnis ersetzt werden sollen. <Anzahl> soll durch die Anzahl der tatsächlich durchgeführten Berechnungsschritte (Vgl. Punkt 6) ersetzt werden.
   Beispiel: Für die Eingabe 0.815 sollte die Ausgabe
   `ln(0.815) = -0.204566, Iterationen: 6`
   sein. **Hinweis:** Die Anzahl der Nachkommastellen spielt keine Rolle.
5. Es dürfen keine Funktionen der Mathematik-Bibliothek (math.h) verwendet werden. Stattdessen soll der natürliche Logarithmus mit Hilfe einer Reihe entwickelt werden (siehe Abschnitt 1.1).
6. Für die Lösung dieser Aufgabe benötigen Sie Schleifen. Die Schleife zur Approximation des natürlichen Logarithmus (siehe Gleichung oben) soll eine endliche Anzahl von Berechnungsschritten durchführen und abbrechen, wenn
   a) entweder eine bestimmte **Genauigkeit** erreicht ist **oder**
   b) eine bestimmte **maximale Anzahl** von Berechnungsschritten durchgeführt wurden.

   Verwenden Sie für die **Genauigkeit** den Zahlenwert 1.0 dividiert durch Ihre Matrikelnummer. Wenn Ihre Matrikelnummer 654321 wäre muss der Zahlenwert für

die Genauigkeit mit $\frac{1.0}{654321} \approx 1{,}5283 \cdot 10^{-6}$ festgelegt werden. Die vorgegebene Genauigkeit ist erreicht, wenn der Absolutbetrag des zuletzt hinzugefügten Summanden der Reihe kleiner als der Zahlenwert für die Genauigkeit ist.
Die maximale Anzahl der Berechnungsschritte soll Ihrer Matrikelnummer Modulo 10000 sein. Wenn Ihre Matrikelnummer 654321 wäre dürfte die Schleife nicht mehr als 4321 Berechnungsschritte durchführen.

Hinweis: Achten Sie darauf, dass die Variablen für die Summe und die Potenzen alle den Datentyp `double` haben! Für alle Berechnungen in Ihrem Programm, bei denen höhere Werte als ca. 2.000.000.000 entstehen können müssen Sie `double` verwenden. Wenn die Programmausgabe des Summenwerts "inf" oder "nan" anzeigt liegt ein numerisches Problem vor, das wahrscheinlich durch einen zu hohen Wert in einer Integer-Variablen (Überlauf) entstanden ist.
Sie sollen die Reihe in jedem Fall **ohne** die Verwendung von mathematischen Funktionen, also nur mit Hilfe von Grundrechenarten programmieren!
Testen Sie Ihr Programm ausführlich mit verschiedenen Eingaben und überprüfen Sie die Ausgaben mit Hilfe eines Taschenrechners.

Folgende Vorgaben muss Ihr Programm zusätzlich einhalten:

- Am Anfang Ihres Programms sollen zwei Kommentarzeilen stehen, in denen der Programm-Titel "Berechnung des natürlichen Logarithmus durch eine konvergierende Reihe" (1. Zeile), Ihr Name (Vor- und Nachnahme) und Ihre Matrikelnummer stehen (2. Zeile). Diese beiden Kommentarzeilen sollen im Programmtext direkt vor der Kopfzeile "int main (void) {" stehen. Es handelt sich nicht um eine Programmausgabe!
- **Alle Variablen in Ihrem Programm sollen am Anfang des Programmtextes definiert werden.** Das ist nach der Kopfzeile des Hauptprogramms ("main") und vor der ersten Programmanweisung. **Die Möglichkeit, eine Variable bei der Definition zu initialisieren darf in Ihrer Lösung nicht genutzt werden!**
- Die Namen der Variablen in Ihrem Programm müssen **mindestens 5 Zeichen lang** sein. Namen, die in den Übungsprogrammen des Moduls 2 vorkommen, z.B. "x_eingabe" oder " meine_matrikelnummer", dürfen nicht verwendet werden. Gleiche aufeinander folgende Zeichen dürfen in den Variablennamen höchstens zweimal vorkommen ("ttt00" wäre nicht erlaubt!). Sie sollen sich Variablen-Namen überlegen, die mit der Verwendung der jeweiligen Variable in Zusammenhang stehen. Insbesondere sollen sich die Unterschiede der Namen zweier Variablen aus deren Verwendung im Programm ableiten lassen. Einfaches Durchnummerieren, z.B. variable1, variable2 ...., ist nicht erlaubt, ebenso wenig sinnfreie Namen wie z.B. torsten, florian oder franziska.
- Ihr Programm darf nur solche Ausgaben erzeugen, die in der Aufgabenstellung explizit gefordert werden.

- Für Ihre Lösung müssen Sie zwei Schleifen programmieren. Dafür sollen Sie in Ihrem Programm das Schleifenkonstrukt `for` oder `while` verwenden, oder beide, je nach dem was für die jeweilige Schleifenaufgabe am besten passt. Die for-Schleife wurde im Praktikumstermin 2 geübt. Die while-Schleife ist in den Vorlesungsunterlagen erklärt oder z.B. hier: http://pronix.linuxdelta.de/C/standard_C/c_programmierung_11_2.shtml#8 .

1.4 Abgabe der Lösung Ihrer Projektaufgabe

Entwickeln Sie Ihr Programm mit Hilfe des "Onlinecompilers",
Webseite: http://CVis.mv.hs-duesseldorf.de .

Wenn Ihr Programm fertig ist und Sie es mit mehreren Eingabewerten getestet haben (Logarithmus-Ergebnisse mit dem Taschenrechner überprüfen!), dann sollen Sie als abschließenden Test mit Ihrem Programm den natürlichen Logarithmus für die folgenden Eingaben durchführen (in der genannten Reihenfolge):

Pos	**Eingabe**	**Beschreibung**
1	1	Genau die linkstehende Eingabe verwenden!
2	0	
3	-1	
4	0.1	
5	73.89056	
6	$0.001 < x < 5 \cdot 10^{-3}$	Es darf ein beliebiger Wert x im genannten Bereich gewählt werden
7	$1 \cdot 10^{5} < x < 1 \cdot 10^{8}$	
8	Erste Stelle Ihrer Matrikelnummer (das wäre die 1 wenn Ihre Matrikelnummer 123456 wäre)	
9	Dritte Stelle Ihrer Matrikelnummer	
10	Vierte Stelle Ihrer Matrikelnummer	
11	Letzte Stelle Ihrer Matrikelnummer (das wäre die 6 wenn Ihre Matrikelnummer 123456 wäre)	

Der Programmlauf des abschließenden Tests muss mit der PDF-Option des Onlinecompilers dokumentiert werden. Dazu müssen Sie vor der Betätigung des RUN-Knopfs die PDF-Option einschalten.

Drücken Sie dazu auf das PDF-Symbol links neben RUN 

. Das PDF-Symbol wird grün. Danach drücken Sie RUN. Der Link auf die PDF-Datei erscheint rechts oben auf der Webseite des Onlinecompilers.
Kontrollieren Sie, ob die vom Onlinecompiler erzeugte PDF-Datei den Programmtext **vollständig lesbar** wiedergibt. Achten Sie insbesondere darauf, dass die einzelnen Programmzeilen nicht zu lang sind. Sie müssen lange Programmzeilen ggf. kürzen bzw. auf zwei oder mehr Zeilen aufteilen. Es gehört mit zur Aufgabe des Projekts, dass Sie das

"Endprodukt", nämlich die vom Onlinecompiler erzeugte PDF-Datei, auf die vollständige Darstellung des Programmtexts prüfen. Wenn Teile des Programms oder der Programmausgaben in dem PDF-Dokument nicht lesbar sind, dann kann das zu **Punktabzug bei der Bewertung** führen.

Die vom Onlinecompiler erzeugte **PDF-Datei** müssen Sie über das Abgabe-Portal "APORT" einreichen (nur PDFs des Onlinecompilers werden benotet!):
https://aport.mv.fh-duesseldorf.de
Dazu benötigen Sie einen **Projektcode**. Der Projektcode für Ihre Abgabe lautet:
PA2_*Gruppenkennung*
Beispiele für Gruppe P5 und Gruppe W2: PA2_P5 bzw. PA2_W2

Der Abgabeschluss für diese Projektaufgabe ist **18 Uhr** des Tages **vor** Ihrem dritten Präsenztermin in diesem Praktikum, also ein Tag vor dem Termin für das Praktikumsmodul 3 für Ihre Gruppe. Die Abgabe über APORT muss von Ihnen persönlich über Ihren persönlichen Shibboleth-Login vorgenommen werden!

Achtung: Bevor Sie eine Lösung für diese Projektaufgabe abgeben, sollten Sie sicher sein, dass Sie in der Lage sind, die Programmierung Ihrer Lösung dem Praktikumsbetreuer erklären zu können, und zwar derart, dass keine Zweifel daran entstehen, dass Sie die Lösung selbständig erarbeitet haben. Bitte beachten Sie den nachfolgenden Abschnitt 2!

Nach der Abgabe Ihrer PDF-Datei im Portal APORT erhalten Sie dort eine elektronische Quittung. Das ist eine Zeichenkette (*Quittungscode*), die Sie sich unbedingt **aufschreiben** und **beim nächsten Praktikumstermin dabei haben** müssen. Sie können mehrmals eine PDF-Datei mit dem gleichen Projektcode einreichen. Als gültige Abgabe zählt immer die zuletzt abgegebene Datei. **Den dazu zugehörenden Quittungscode (und den Projektcode!) müssen Sie zum nächsten Praktikumstermin mitbringen.**

2 Wichtige Regeln für die Bearbeitung, die Abgabe und die Bewertung von Projektaufgaben

Mit den Projektaufgaben sollen Sie lernen, C-Programme mit den bisher geübten Sprachmitteln zu entwickeln und zu testen. Sie sollen auch lernen, eine Software-Aufgabenbeschreibung sorgfältig zu lesen, zu verstehen und in eine vollständige Lösung umzusetzen. Verlassen Sie sich bei Verständnisfragen zu den einzelnen Punkten der Aufgabenstellung niemals auf die Aussagen von Personen, die die Aufgabenstellung eventuell nicht richtig interpretieren können oder nicht vollständig kennen. Gültig ist nur der Text der Aufgabenstellung. Es gibt ausreichend Gelegenheit um eventuelle Verständnisfragen mit einem Betreuer, einem Tutor oder Prof. Zielke zu klären. Bei der Bewertung wird das

Argument "konnte man nicht verstehen" nicht akzeptiert. Sie hätten vorher nachfragen können.

Sie sollen diese Projektaufgaben selbstständig bearbeiten und lösen. Alle Lösungen, die aus der Betrachtung des Quellcodes und dem Vergleich mit anderen abgegeben Lösungen vermuten lassen, dass die Bearbeitung nicht eigenständig durch Sie selbst erfolgte, werden mit 0 Punkten bewertet oder zumindest stark abgewertet. Diese Abwertung betrifft auch diejenigen Lösungen, die als Quelle für andere abgegebene Lösungen vermutet werden können.
Achtung: Wenn Ihre Lösung von anderen Praktikumsteilnehmern verwendet wird, müssen Sie mit dem Verlust der Bewertungspunkte rechnen, ggf. auch nachträglich. Bei offensichtlichen Täuschungsversuchen behalten wir uns vor, die gesamte Praktikumsteilnahme mit 0 Punkten zu bewerten.
Eine Bewertung Ihrer Lösung ist generell nur möglich, wenn die Abgabe fristgerecht mit Ihrem persönlichen Zugang über APORT erfolgt ist, und wenn Sie an dem jeweils nächsten Praktikums-Präsenztermin **Ihrer** Gruppe teilnehmen!

Mit der Abgabe einer Lösung für diese Projektaufgabe erklären Sie sich automatisch mit den hier formulierten Bewertungsprinzipien einverstanden.