Abteilung für Angewandte Mathematik — Ludwig Striet — SS 2021 Hedwig Keller

Einführung in die Programmierung für Studierende der Naturwissenschaften

Blatt 11 – 05.07.2021

Abgabe: *14.7.2021 bis 18 Uhr per e-Mail an Ihre:n Tutor:in*

Aufgabe 1 ((5 Bonuspunkte)). Schreiben Sie eine C++-Funktion

```
bool contains(std::string str1, std::string str2,
  size_t* pos = NULL)
{
  // ...
}
```

die Überprüft, ob `str2` in `str1` enthalten ist. (Zum Beispiel ist EI in BEISPIEL enthalten, SPIELT jedoch nicht.) Falls der string enthalten ist und `pos` nicht `NULL` ist, soll in `*pos` die Position geschrieben werden.

Aufgabe 2 (5 Bonuspunkte). Das Bisektionsverfahren zur Approximation einer Nullstelle einer gegebenen Funktion $f\colon \mathbb{R} \to \mathbb{R}$ ist wie folgt definiert:

(1) Wähle eine Genauigkeit $\varepsilon > 0$ und zwei Startwerte $a_0, b_0 \in \mathbb{R}$, sodass $a_0 < b_0$ und $f(a_0)$ und $f(b_0)$ unterschiedliches Vorzeichen haben. Setze $k = 0$.
(2) Falls $b_k - a_k < \varepsilon$, so ist $[a_k, b_k]$ das gesuchte Lösungsintervall: Breche das Verfahren mit Ausgabe $\frac{a_k+b_k}{2}$ ab.
(3) Falls $f(\frac{a_k+b_k}{2})$ das gleiche Vorzeichen wie $f(a_k)$ hat, setze $a_{k+1} = \frac{a_k+b_k}{2}$ und $b_{k+1} = b_k$. Andernfalls setze $a_{k+1} = a_k$ und $b_{k+1} = \frac{a_k+b_k}{2}$.
(4) Erhöhe k um 1 und gehe zu (2).

Implementieren Sie dieses Verfahren in OCTAVE als Funktion

`function nst = bisektion(f,a,b,eps),`

wobei die als Parameter übergebene Funktion `f` ein sogenannter *function handle* ist, d.h. eine Variable, welche den Verweis auf eine Funktion speichert. Die Syntax zur Definition solcher auch als *anonyme Funktion* bezeichneter Variablen ist dabei wie im folgenden Beispiel, in dem die Funktion `f(x)` den Wert von $\mathtt{x}^2 - 1$ berechnet:

```
f = @(x) x*x-1;
```

Der Rückgabewert `nst` Ihrer Funktion sollte der Mittelpunkt des Lösungsintervalls sein. Testen Sie Ihr Programm, indem Sie damit die Nullstelle der Funktion $x \mapsto x^3 - 2$ approximieren.

Aufgabe 3 (5 Bonuspunkte)**.** Verwenden Sie OpenMP um eine Funktion

```
1       int parallel_min(int* L, unsigned int n, unsigned int np)
2       {
3         // ...
4       }
```

zu schreiben, die `np` Threads benutzt, um das Minimum in einer Liste `L` der Länge `n` zu finden.

Aufgabe 4 (5 Bonuspunkte)**.** Passen Sie den Code aus der Vorlesung für das Merge Sort Verfahren so an, dass es mit vier Threads eine Liste beliebiger Länge sortiert. Als Grundlage können Sie die Datei `merge_sort.cc` von der Vorlesungswebseite verwenden. Verwenden Sie dazu OpenMP.

Aufgabe 5 (optional)**.** Fügen Sie Ihrer Abgabe eine Datei `erfahrungen11.txt` bei. Berichten Sie darin wieder in Stichpunkten bzw. ein bis zwei kurzen Sätzen über Ihre Erfahrungen mit Kursinhalt und Übungsaufgaben. Was fiel Ihnen leicht? Was ist noch unklar? Wie viel Zeit haben Sie für die Bearbeitung der Hausaufgaben benötigt und welche Probleme traten dabei auf?