Nachname: ..... Vorname: ..... MatrNo.: .....

Klausur: Programmieren 2

Hochschule für Angewandte Wissenschaften Hamburg
Fakultät Technik und Informatik, Department Informations- und Elektrotechnik
Prof. Dr. Robert Heß, 27.1.2017, Dauer: 180 Min.
Erlaubte Hilfsmittel: Vorlesungsunterlagen, Lösungen aus dem Praktikum und C/C++ Einführungsbücher.

Ergebnis: ..... von 100 Punkten Note: ..... Punkte.

---

1 Einführung

Die *Fibonacci-Folge* ist eine unendliche Folge natürlicher Zahlen, mit der Leonardo Fibonacci 1202 das Wachstum von Kaninchenpopulationen beschrieb. Die Folge findet sich in zahlreichen anderen Wachstumsprozessen wieder und war schon in der Antike bekannt.

Die Elemente der Fibonacci-Folge, die *Fibonacci-Zahlen*, lassen sich auf unterschiedliche Weise berechnen. Die wohl bekannteste Methode bestimmt den Wert einer Fibonacci-Zahl f_n rekursiv aus der Summe der beiden vorigen $f_{n-2} + f_{n-1}$, wobei die ersten beiden Zahlen f_1 und f_2 mit eins definiert sind (1). Eine weitere Methode bestimmt direkt den Wert einer Fibonacci-Zahl f_n mittels einer etwas aufwändigeren Formel (2). Schließlich lassen sich die Fibonacci-Zahlen exponentiell nähern (3).

$$\text{rekursiv:} \qquad f_n = f_{n-2} + f_{n-1} \qquad f_1 = f_2 = 1 \tag{1}$$

$$\text{direkt:} \qquad f_n = \frac{1}{\sqrt{5}} \left[\left(\frac{1+\sqrt{5}}{2} \right)^n - \left(\frac{1-\sqrt{5}}{2} \right)^n \right] \tag{2}$$

$$\text{genähert:} \qquad f_n \approx e^{a\,n+b} \qquad a = \operatorname{arsinh}(1/2) \qquad b = -\frac{\ln(5)}{2} \tag{3}$$

In dieser Aufgabe sollen die Fibonacci-Zahlen nach diesen drei Methoden berechnet, in einen dynamischen Vektor gespeichert, und anschließend in eine CSV-Datei geschrieben werden. Folgende Funktionen aus der Bibliothek *math.h* werden benötigt:

sqrt(x)	Quadratwurzel $\sqrt{x}$
pow(x, y)	Potenz x^y
exp(x)	Exponentialfunktion $\exp(x) = e^x$
asinh(x)	Areasinus Hyperbolicus $\operatorname{arsinh}(x)$
log(x)	natürlicher Logarithmus $\ln(x)$

2 Programmieraufgaben

Aufgabe 1 (10 Punkte)

Erstellen Sie einen Datentyp mit Namen *tFibonacci* für eine Struktur mit den vier Elementen *index* (int), *rekursiv* (double), *direkt* (double) und *genaehert* (double). Legen Sie im Hauptprogramm einen Zeiger auf diesen Datentyp an.

Aufgabe 2 (25 Punkte)

Erstellen Sie eine Funktion mit Namen *ErstelleTabelle* die als einzigen Parameter die *Anzahl* der Elemente in der Tabelle erwartet und einen Zeiger auf den dynamisch reservierten Speicher für die Tabelle zurückgibt.

Die Funktion reserviert zunächst Speicher für die Tabelle. Danach wird die Tabelle mit den Fibonacci-Zahlen f_1 bis f_{Anzahl} gefüllt. Jede Zeile der Tabelle soll gemäß der erstellten Struktur folgende Elemente haben:

- Index n der aktuellen Fibonacci-Zahl f_n.
- Rekursiv berechnete Fibonacci-Zahl f_n gemäß Formel (1). Um die CPU nicht unnötig zu belasten, soll keine rekursive Funktion erstellt werden, sondern es sollen ab der dritten Zeile die bereits berechneten Zahlen f_{n-1} und f_{n-2} verwendet werden.
- Direkt berechnete Fibonacci-Zahl f_n gemäß Formel (2).
- Genäherte Fibonacci-Zahl f_n gemäß Formel (3).

Bei Erfolg liefert die Funktion die Adresse des dynamisch reservierten Vektors, bei Misserfolg die Adresse *NULL* zurück.

Aufgabe 3 (10 Punkte)

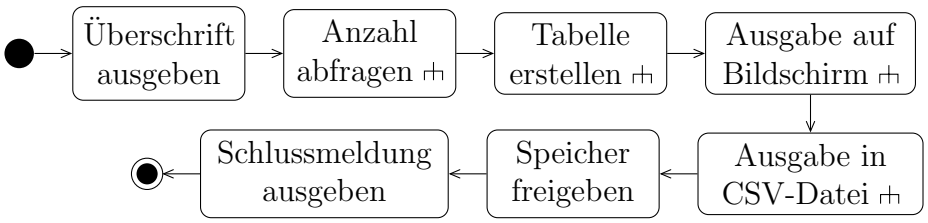
Erstellen Sie eine Funktion mit Namen *AusgabeBildschirm*, welche die Tabelle mit ihren vier Spalten ordentlich ausgerichtet auf dem Bildschirm ausgibt. Die Funktion gibt keinen Wert zurück.

Aufgabe 4 (15 Punkte)

Erstellen Sie eine Funktion mit Namen *AusgabeCsv*, welche die Tabelle in eine CSV-Datei mit Namen *Fibonacci.csv* schreibt. Die Funktion schreibt in die erste Zeile den Text „Fibonacci-Zahlen“ und in die zweite Zeile die Spaltenüberschriften *Index*, *rekursiv*, *direkt* und *genähert*. Danach werden alle Zeilen der Tabelle entsprechend der Überschriften geschrieben. Verwenden Sie ein Komma als Listentrennungszeichen (kein Semikolon). Die Funktion gibt bei Erfolg eine null und sonst einen Wert ungleich null zurück.

Aufgabe 5 (15 Punkte)

Fügen Sie die Programmstücke gemäß folgendem Aktivitätsdiagramm zusammen, wobei Sie die Anzahl der Zeilen in der Tabelle mit einer sinnvollen Funktion vom Benutzer abfragen:



Achten Sie auf eine gute Benutzerführung, fangen Sie mögliche Fehleingaben des Benutzers ab, vermeiden Sie globale Variablen und entfernen Sie alle Fehler und Warnungen des Compilers.

3 Schriftliche Aufgaben

Aufgabe 6 (3 Punkte)

Warum sollten in C-Programmen Sprünge mit *goto* vermieden werden?

Aufgabe 7 (3 Punkte)

Was ist das charakteristische Merkmal einer rekursiven Funktion?

Aufgabe 8 (8 Punkte)

a) Definieren Sie eine Struktur mit den beiden Elementen x und y jeweils vom Typ *float*.

b) Definieren Sie zur der eben erstellten Struktur einen Datentyp mit Namen *tPoint*.

c) Definieren Sie zu dem neuen Datentyp eine Variable a und einen Zeiger pa und lassen Sie pa auf a zeigen.

d) Initialisieren Sie mittels pa die Elemente von a mit null unter Verwendung der sinnvollsten Syntax.

Aufgabe 9 (3 Punkte)

Sie wollen mittels einer *int*-Variable mit den Werten 1 bis 5 auf fünf unterschiedliche Optionen verzweigen. Welche Syntax bietet sich am sinnvollsten an?

☐ *if*-Verzweigung ☐ *else-if*-Verzweigung ☐ *switch-case*-Verzweigung

Aufgabe 10 (8 Punkte)

Die Elemente einer einfach verketteten Liste haben für die Verkettung einen Zeiger mit Namen *next*. Erstellen Sie den Quellcode, mit dem ein neues Element, auf das der Zeiger *pNeu* zeigt, hinter das Element eingefügt wird, auf das der Zeiger *pAktuell* zeigt.