"Informatik II"

2013

1. Dynamische Datenstrukturen (32 Punkte)
   Gegeben sei folgendes C++-Programm:

```
#include <iostream>

using namespace std;

struct lelem {
    int data;
    lelem * pNext;
};
main()
{
  lelem * anchor, * lastitem, * newitem;
  int i=0,j;

  anchor = new lelem;
  lastitem = anchor;
  lastitem -> data = i;
  lastitem -> pNext = lastitem;

  for (i=1; i<10;i++)
    {
      newitem = new lelem;
      newitem -> pNext = newitem;
      lastitem -> pNext = newitem;
      newitem -> data = 7 % i;
      lastitem = newitem;
    }
}
```

   a. Stellen Sie die Datenstruktur grafisch dar, die in diesem Programm aufgebaut wird (inklusive aller vorkommenden Zeiger nach Abarbeitung des Programms und der Datenwerte). (6 Punkte)

b. Ergänzen Sie das Programm um eine Funktion

```
int del_elem(lelem *& anch, int del_data);
```

mit folgendem Ablauf:

I. Suchen des ersten Elementes mit dem Datenelement **del_data** in der Datenstruktur, die durch den Pointer **anch** adressiert wird.

II. Wenn in I. ein Element gefunden wurde, so ist dieses Element aus der Datenstruktur zu löschen, der von diesem Element benutzte Speicherplatz soll freigegeben werden. Zurückzugeben werden soll der Wert 1 (Löschen ist erfolgt) bzw. 0 (sonst).
Hinweis: Beachten Sie die folgenden Sonderfälle:

i. Es wird die leere Liste übergeben

ii. Die Liste enthält nur ein Element und dieses wird gelöscht.

iii. Das erste Element wird gelöscht.

(14 Punkte)

c. Ergänzen Sie das Programm um eine Funktion

```
int print_elem(lelem *anch, int anz);
```

mit folgender Aufgabe:
Es sollen alle Datenelemente der Datenstruktur, die durch den Pointer **anch** adressiert wird, durch Tabulatoren getrennt auf dem Bildschirm ausgeben werden. Dabei sollen, wenn möglich, **anz** Elemente auf einer Bildschirmzeile stehen.

d. Demonstrieren Sie die Verwendung der Funktionen **del_elem** und **print_elem** im Hauptprogramm. (5 Punkte)

Hinweis: in den Teilaufgaben b - d ist nicht davon auszugehen, dass die Anzahl der Elemente in der Datenstruktur bekannt ist.

2. Objektorientierte Programmierung (23 Punkte)
   Geben Sie eine Klassendefinition für eine Klasse **kreis** an, welche einen privaten Member (Datenelement) radius vom Typ float verwaltet. Folgende Methoden sollen in der Klasse enthalten sein:

   - Konstruktor mit keinem Parameter, der Member soll mit 0 initialisiert werden (Standardkonstruktor)
   - Konstruktor mit einem Parameter vom Typ float. Der Member soll mit dem Werten des Parameters initialisert werden
   - Methode set mit einem Parameter. Diese Methode setzt den Wert des Members mit dem Wert des Parameters.
   - Methode flaeche. Diese Methode gibt den Wert der Fläche des durch den privaten Member beschriebenen Kreises zurück.
   - Methode flaeche_sektor mit einem Parameter vom Typ float. Diese Methode gibt den Wert der Fläche des durch den privaten Member und dem Parameter, welcher als Winkel α interpretiert werde n soll, beschriebenen Kreissegmentes zurück.
   - Methode umfang. Diese Methode gibt den Wert des Umfanges des durch den privaten Member beschriebenen Kreises zurück.
   - Methode print. Diese Methode gibt den Wert des Members auf den Bildschirm aus.

   Geben Sie die Implementierung der Methoden an (gegebenenfalls inline-Code). Demonstrieren Sie die Verwendung der Klasse **kreis** in einem kurzen Hauptprogramm, in dem alle Methoden verwendet werden.
   Hinweis: Es gelten folgende Formeln:
   Kreisfläche: $f = \pi r^2$
   Kreisumfang: $u = 2\pi r$
   Kreissegment: $s = \frac{\pi r^2 \alpha}{360}$
   Pi: $\pi = 3.141592653589793$