Dr. M. Grabert
Johannes Mayer

Objektorientierte Programmierung mit C++ (WS 2001/2002)

Abgabetermin: 8. November 2001

Beispiellösung

2 Sortieren mit Struktur (4 Punkte)

Verwenden Sie das Sortier-Template aus der Vorlesung, d. h. binden Sie die Header-Datei *sort.h* ein, und schreiben Sie ein kleines Programm, welches dieses Template verwendet. Dieses Programm soll Strukturen, die Name, Vorname und Geburtsdatum einer Person enthalten, nach dem Namen sortieren. (So ähnlich wie *main2.cc* nur eben mit Strukturen, die Charakter-Pointer enthalten.)

Lösung:

structsort.cc

```
#include "sort.h"
#include <stdio.h>

struct person {
   char *name;
   char *vorname;
   char *gebdat;
} pers[] = {
{ "Matthias", "Grabert", "26.05.1832"},
{ "Franz", "Gans", "37.12.2002" },
{ "Heini", "Keini", "11.11.1111" }
};

int perscmp(person a, person b)
{
   return strcmp(a.name, b.name);
}
```

```
int main()
{
   int len = sizeof(pers)/sizeof(*pers);

   bubblesort(pers, len, perscmp);
   for (int i = 0; i<len; i++)
      puts(pers[i].name);

   return 0;
}
```

3 Nur rauf damit (8 Punkte)

Na, Templates sind doch schön, oder? Sind Sie etwa nicht so recht davon überzeugt? Dann ist es wohl besser, wenn wir etwas damit umgehen. Schreiben Sie doch einfach :-) eine Header-Datei `stack.h`, in der Sie einen Stack mittels Templates implementieren. Um die Schnittstelle müssen Sie sich gar nicht erst kümmern, denn die beiden Programme *ints.cc* und *strings.cc* sollen beide mit Ihrer Header-Datei `stack.h` laufen. Die Implementierung des Stacks soll für eine beliebige Größe ausgelegt sein! Also nichts mit Array, oder so! Implementieren Sie den Stack als lineare Liste selbst (ohne Verwendung der STL ;-()!
Wenn sie sich die Programme *ints.cc* und *strings.cc* ansehen, wird Ihnen etwas neues auffallen. Es gibt nämlich nicht nur Funktionstemplates, sondern auch Struktur-Templates. Man deklariert und verwendet ein Struktur-Template wie folgt:

```
template <class T>
struct S {
T var;
};
...
int main() {
S<int> a;
a.var = 3;

S<char*> b;
b.var = "Test";
}
```

Die Datenstruktur für ein Element der linearen Liste ist ein Struktur-Template. Nun kanns aber los gehen ... mit den Templates!

Lösung:

stack.h

```
#ifndef __STACK_H
#define __STACK_H
```

```
#include <stdlib.h>

template <class T>
struct Stack {
   Stack *next;
   T element;
};

template <class T>
void newStack(Stack<T> *&s) {
   s = NULL;
}

template <class T>
void deleteStack(Stack<T> *&s) {
   if (s != NULL) {
      deleteStack(s->next);
      free(s);
      s = NULL;
   }
}

template <class T>
void push(Stack<T> *&s, T element) {
   Stack<T> *elem = (Stack<T>*) malloc(sizeof(Stack<T>));

   elem->next = s;
   elem->element = element;

   s = elem;
}

template <class T>
T pop(Stack<T> *&s) {
   if (s != NULL) {
      Stack<T> *elem = s;
      s = elem->next;
      T element = elem->element;
      free(elem);
      return element;
   }
}

template <class T>
int isEmpty(Stack<T> *&s) {
   return s == NULL;
}

#endif
```

4 Chat mal wieder (8 Punkte)

Sie betreten einen Chatroom. Sie schlüpfen für den Chat in eine neue virtuelle Persönlichkeit. Der Rechner fragt Sie, wen Sie darstellen wollen. „Ganz gleich!“, klicken Sie an. Der Rechner sucht Ihnen automatisch eine neue Identität ... und nach kurzem Warten kommt er rüber: „Sie heißen von nun an 'Ingo Melzer' und tragen seine Aufgaben bezüglich SLC mit sich rum“. Da kommt schon der erste Mitchatter auf Sie zu: „Hey! Spuck mal alle Namen von Leuten aus, die bereits mehr als 100 Seiten in dem Semester gedruckt haben.“ „Easy, man - ich machs in Perl!“, antworten Sie. „Nein, Scheisserchen! Machs mal in C++ mit Map-Templates. Und ich drück Dir 8 Punkte auf Dein Konto“, antwortet Ihr virtuelles Gegenüber. Beeindruckt von des anderen Persönlichkeit (eine gewisse „Mattina“) und Ausdrucksgabe beschließen Sie, der Aufforderung Folge zu leisten.
Wir, von der SAI, helfen Ihnen natürlich in allen Lebenslagen, auch in den virtuellen! Wir verraten Ihnen: In einer *SLC-Tabelle* (=Datei) stehen die UserIds und RealNames der Rechnerbenutzer drin (durch Doppelpunkt getrennt):

```
14460:Carolina Gruenschloss
13503:Stefan Kraus
14391:Jan-Hendrik Otte
...
```

In einer *2. Tabelle* werden alle Druckjobs erfaßt. Aufbau: UserId und Anzahl der gedruckten Seiten in dem Druckjob, wieder durch Doppelpunkt getrennt:

```
14460:5
13503:22
14460:12
14391:85
...
```

Schreiben Sie ein reelles C++-Programm, das unter Verwendung der in den Übungen gezeigten Map-Templates die RealNames und verbrauchten Druckseiten aller Teilnehmer ausgibt, die bereits mehr als 100 Seiten gedruckt haben. Das Programm soll mit den 2 Dateinamen als Parametern aufgerufen werden:

```
printcheck pwd_file print_file
```

Lösung:

druckcheck.cc

```
#include <stdio.h>
#include <strings.h>
#include <string>
#include <map>

#define LIMIT 100
```

```
int main(int argc, char **argv)
{
   if (argc<3)
      fprintf(stderr, "%s: pwd druckseiten\n", argv[0]),
      exit(1);

   FILE *pwd = fopen(argv[1], "r");
   if (!pwd)
      perror(argv[1]),
      exit(1);

   FILE *print = fopen(argv[2], "r");
   if (!print)
      perror(argv[2]),
      exit(1);

   map<int, string>id_name;   // UID = Index, Voller Name
   map<int, int>id_seiten;    // UID und gedruckte Seiten

   char buf[1024];
   while (fgets(buf, sizeof(buf), pwd)) {
      buf[strlen(buf)-1] = 0;
      char *pos = strchr(buf, ':');
      if (!pos)
         fprintf(stderr, "wrong line pwdfile: '%s'\n", buf),
         exit(1);
      *pos = 0;
      int uid = atoi(buf);
      string hlp = pos+1;
      id_name[uid] = hlp;
   }
   while (fgets(buf, sizeof(buf), print)) {
      buf[strlen(buf)-1] = 0;
      char *pos = strchr(buf, ':');
      if (!pos)
         fprintf(stderr, "wrong line printfile: '%s'\n", buf),
         exit(1);
      *pos = 0;
      int uid = atoi(buf);
      int seiten = atoi(pos+1);
      id_seiten[uid] += seiten;
   }
   int ges = 0;
   for (map<int,string>::iterator i = id_name.begin();
               i != id_name.end(); i++) {
      ges ++;
      if (id_seiten[i->first] >=LIMIT)
         cout << id_name[i->first] << "\n";
   }
```

```
  fclose(print); fclose(pwd);
  cout << ges << " User insgesamt haben mehr als " << LIMIT <<"Seiten.\n";
}
```