Beispiel 1: Strings (48 Punkte)

Entwerfen Sie eine Klasse namens String, die einen privaten Zeiger auf einen dynamischen Speicherbereich (private: char *string) besitzt. Als weitere Attribute sollen len für die Stringlänge (unsigned long len) und bufsize (unsigned long bufsize) für die Größe des Stringbuffers angelegt werden.

Schreiben Sie drei Konstruktoren String(void), String(const char*) und String(const char), die den nötigen Speicherbereich reservieren, und zwar soll der für den String verwendete Speicherbereich immer um 15 Zeichen größer sein als benötigt.

Schreiben Sie drei Zuweisungsoperatoren operator=(String), operator=(const char*) und operator=(const char) sowie einen copy-Konstruktor String(const String&). Vergessen Sie den Destruktor (~String()) nicht.

Um die vom Benutzer ansprechbaren Funktionen von der tatsächlichen Verwaltung des Speichers zu kapseln, entwerfen Sie eine private Methode replace (void replace(const char*)), die einen neuen String übernimmt und alle dazu nötigen Maßnahmen ergreift.

Des weiteren soll die Klasse mit überladenen <<- und >>-Operatoren ausgestattet werden, wobei die Eingabe vorerst mit einer maximalen Größe arbeiten darf. (friend ostream &operator<<(ostream&, const String&) und friend istream &operator>>(istream&, String&))

Implementieren Sie für operator+ und operator += alle nötigen Funktionen, um die Datentypen const String&, const char* und const char verarbeiten zu können.

Überlegen Sie, welche Funktionen als Elementfunktionen deklariert werden können und welche als Nicht-Elementfunktionen deklariert werden müssen.

Überladen Sie den Operator[], um auf die einzelnen Zeichen des String-Objekts genau so zugreifen zu können wie auf ein char-Feld.

Benutzen Sie zur Implementierung der operator-Funktionen eine Funktion insert(unsigned long pos, unsigned long len, const char*), die an der Stelle pos im Stringspeicher die ersten len Zeichen des Strings s einfügt. insert soll dabei Gebrauch vom zusätzlichen Speicher des Stringpuffers machen. Schreiben Sie eine Methode Overwrite für const String& und const char*, die den String ab der übergebenen Position mit dem übergebenen String überschreibt. Gegebenenfalls muß der String verlängert werden.

Wichtig:

- Trennen Sie die Deklarationen und Definitionen voneinander, und speichern Sie diese in zwei Dateien namens String.h und String.cpp.
- Implementieren Sie Ihre Klasse innerhalb eines Namespace. Die Namesgebung ist für Sie frei wählbar.
- Alle Ausnahmesituationen (Bereichsüberschreitungen, Allokationsfehler, etc), die auftreten könnten sind mittels exception handling (try, throw, catch) abzufangen!!! Z.B. try {.....if(!string) throw(Allokationsfehler()); } catch(Allokationsfehler) {.....}

Abgabe: bis 18.1.2000 23:59 Uhr elektronisch (Verzeichnis uebung7_8 einrichten!) und am 19.1. 2000 zu Beginn des Praktikums auf Papier (Listing der Quelldateien).