Universität Potsdam
Institut für Informatik

Praxis der Programmierung

7. Aufgabenblatt

1. Kopieren Sie die Datei `rec_count.c` und compilieren Sie sie. Ergänzen Sie die Datei so, dass sie ohne Warnungen übersetzt wird. Modifizieren Sie nun die Implementierung der Funktion `rec_out`, so dass bei Ausführung von `rec_count` folgende Ausgabe entsteht:

```
Die 1. Ausgabe.
Die 2. Ausgabe.
Die 3. Ausgabe.
Die 4. Ausgabe.
Die 5. Ausgabe.
Die 6. Ausgabe.
```

2. Schreiben Sie zwei Header-Dateien `date.h` und `highscore.h`, die die Definitionen der Typen `Date` bzw. `Highscore` enthalten. Außerdem sollen jeweils die Signaturen von zwei Funktionen bekanntgegeben werden (ohne Implementierungen).

   - Dabei ist `Date` ein Strukturtyp, der drei `int`-Variablen für einen Tag, einen Monat und ein Jahr als Member bereitstellt. Es wird vereinbart, dass eine Funktion zum Setzen eines Datums auf den 1.1. eines Jahres existiert, wobei das Jahr as Parameter übergeben wird. Außerdem soll es eine Funktion geben, die zur sinnvoll formatierten Ausgabe eines Wertes vom Typ `Date` dient.
   - `Highscore` ist ein Strukturtyp, der ein Datum vom Typ `Date` und einen Scorewert vom Typ `int` als Member bereitstellt. Variablen vom Typ Highscore repräsentieren einen Highscore-Stand eines Spiels, der am Tag des Datums erreicht wurde. Es werden zwei Funktionen deklariert: eine zum Setzen eines Wertes vom Typ `Highscore` und eine zur geeigneten formatierten Ausgabe eines solchen Wertes.

3. Schreiben Sie zwei Dateien `date.c` und `highscore.c`, die die Implementierungen der in den Header-Dateien vereinbarten Funktionen enthalten. Außerdem besitzt die Datei `highscore.c` eine `main`-Funktion zum Testen der Funktionen von `highscore.c`.

4. Entwickeln Sie eine Anwendung `scores.c`, die alle Funktionen von `highscore.c` testet. Welcher Compiler-Fehler tritt auf, wenn Sie `highscore.c` nicht anpassen?

5. Um die Fehlermeldung zu vermeiden, aber weiterhin die Datei `highscore.c` schnell testen zu können, ergänzen Sie Präprozessordirektiven, die die Übersetzung der `main`-Funktion von `highscore.c` ein- oder ausschalten können und schalten Sie damit deren Übersetzung ab.

6. Ergänzen Sie in `highscore.h` die Deklaration (ohne Initialisierung!) einer Variablen `firstYear` vom Typ `int`, um in `.c`-Dateien das Jahr festzulegen, in dem die Aufzeichnung der Highscorewerte begonnen hat. Modifizieren Sie die Funktion zur formatierten Ausgabe von Highscorewerten in `highscore.c` so, dass immer zuerst ausgegeben wird:

   ```
   Scores ab XXXX:
   ```

   wobei für `XXXX` der Wert von `firstYear` einzusetzen ist.

   Setzen Sie in der Testapplikation `scores.c` den Wert dieser Variablen auf 2017 und testen Sie, ob das veränderte Verhalten erreicht wurde.

   Wie stellen Sie am besten sicher, dass die Testanwendung eine konsistente Ausgabe erzeugt?

   ---

7. Kopieren Sie die Datei `rechte.c` und analysieren Sie den Quellcode. Ergänzen Sie ihn so, dass die Rechte in Abhängigkeit des Wertes von `umask` für reguläre Dateien und für Verzeichnisse so berechnet werden, wie im UNIX-Dateisystem.
   Verwenden Sie ausschließlich Bitoperationen!

8. Schreiben Sie ein C-Programm, das eine Funktion

   ```
   unsigned long long explode(unsigned int m, unsigned short n)
   ```

   definiert und testet, die *nur unter Verwendung von Bitoperationen* den Wert $m \cdot 2^n$ berechnet und zurückgibt.

   Hinweis: Verwenden Sie die Formatlemente `%u` *und* `%llu`*.*