Bedeutung und Beweis von C-Programmen

Fragenkatalog zur Prüfung, Uni Bremen, SoSe 08

Christoph Lüth 1. August 2008

Vorbemerkung

Zur Prüfung erlaubte Unterlagen:

- Der C-Standard (ISO IEC 8988:1999, wird gestellt)
- Vorlesungsnotizen und Folienkopien von der Webseite
- Eigene Mitschriften

Es geht bei der Prüfung *nicht* darum, Begriffe aus dem Standard *auswendig* zu lernen, daher bringe ich den C-Standard zur Prüfung mit. Wir wollen sehen, dass ihr mit dem Standard umgehen könnt, und wisst, wo man was findet, nicht dass ihr alle Begriffe auswendig gelernt habt. Wenn man allerdings bei Frage 13 anfängt, auf Seite 1 zu blättern, wäre das wiederum ein schlechtes Zeichen.
Bezüglich Bedeutung und Beweis soll auch nicht geprüft werden, ob ihr die Regeln alle auswendig könnt, sondern es soll geprüft werden, ob ihr die Grundprinzipien der Typableitung, Semantik und der Hoare-Regeln verstanden habt.

Fragenkatalog

Der Standard

1. Was genau *definiert* der Standard?
2. Was bedeuten die Begriffe *undefiniert*, *unspezifiziert* und *implementationsabhänging*? Was sind Beispiele dafür?

Die Sprache: Typen

3. Welche Typen gibt es in C?
4. Welche Typkonversionen gibt es? Wie sind sie spezifiziert?
5. Was ist der *integer conversion rank*?
6. Welches sind die Unterscheide zwischen Zeigern und Feldern? Wann werden Felder in Zeiger konvertiert, wann nicht?
7. Gegeben folgende Deklarationen

   ```
   int x; unsigned long y;
   ```

   was ist der Typ des Ausdrucks (und warum):

   ```
   x+y, sin(x)*y
   ```

8. Was ist ein *object*? Was ist ein *incomplete type*? Was ist ein *qualified type*?

9. Wann sind zwei Typen kompatibel?
10. Was ist ein: *declarator*; *direct-declarator*; *type qualifier*; *storage class*.
11. Mit welchen Präzedenzregeln werden Deklarationen parsiert?
12. Wie wird folgende Deklaration parsiert, und was bedeutet sie:

```
void (*signal(int sig, void (*func)(int)))(int);
```

Die Sprache: Ausdrücke

13. Was ist ein *lvalue*?
14. Was ist ein *sequence point*, und wo treten welche auf?
15. Was gibt es für Unterscheide bei Funktionen mit und ohne Prototypen?
16. Was ist der Unterschied zwischen `a++`, `a += 1`, `a= a+1`?
17. Welche verschiedene Fälle unterscheidet die Semantik für Multiplikation und Subtraktion?
18. Auf welche Typen kann die Gleichheitsrelationen (`==`, `!=`), und auf welche die Vergleichsrelationen (`<`, `<=`, ...) anwenden? Welches sind die mögliche Ergebnisse?
19. Gibt ist einen Unterschied zwischen `x && y` und `! ((0 == x)||(0== y))`?
20. Welche Operatoren sind C strikt, und welche nicht (und was bedeutet das)?
21. Welche Einschränkungen gibt es bei der Zuweisung `x= a` an die Typen von `x` und `a`?

Bedeutung

22. Wie formalisieren wir die Typableitung?
23. Wie modellieren wir den Speicher?
24. Mit welchen Funktionen haben wir die denotationale Semantik der Sprache modelliert? Was ist ihre Signatur, und welche Bedeutung haben Argumente und Rückgabewerte?
25. ... und was ist überhaupt "denotationale Semantik"?
26. Welche Eigenschaften der Sprache haben wir modelliert, welche nicht?

Beweis

27. Wie funktioniert der Floyd-Hoare-Kalkül?
28. Warum funktioniert der Flody-Hoare-Kalkül aus der Literatur nicht direkt für C? Welche Änderungen mussten wir vornehmen?
29. Welche Spracheinschränkungen haben wir für den Beweiskalkül vorgenommen, und warum?
30. Was für Eigenschaften können wir mit dem unserem Kalkül beweisen?

Rechtlicher Hinweis: Dieser Fragenkatalog begründet keinen Ausschließlichkeitsanspruch. Die Veranstalter behalten sich das Recht auf andere, den Stoff der Veranstaltung betreffende, Fragen vor.