GTI, FORMALE LOGIK UND VERIFIKATION SS 2017

Prof. Dr. Sebastian Iwanowski

Übungsblatt 06 (6 Aufgaben) S.1/4

fhwedel UNIVERSITY OF APPLIED SCIENCES

Aufgabe 1)

Das folgende Programm berechnet eine natürliche Potenz von b deutlich schneller als die einfachere Methode, n mal b mit sich selbst zu multiplizieren. Hier soll nicht bewiesen werden, dass das schneller geht, sondern nur dass das Programm überhaupt das gewünschte Ergebnis liefert.

Die Geschwindigkeitsgewinn kann für große n leicht durch ein eigenes Programm nachgeprüft werden. Hinweis: Arbeiten Sie in herkömmlichen Sprachen wie Pascal nicht mit Integerwerten, weil Sie sonst MAXINT überschreiten.

```
{ b ∈ Z, e ∈ N }                                          φ

   result := 1;
   base := b;
   exp := e;

   while exp > 0
     begin
       if (exp MOD 2 = 1) then
         begin
           result := result * base;
           exp := exp - 1;
         end
       else
         begin
           base := base * base;
           exp := exp DIV 2;
         end;
     end; {while}
{ result = b^e }                                          ψ
```

a) Beweisen Sie mit vollständiger Induktion über k: $b^e = base_k^{\exp_k} \cdot result_k$
(Hierbei sind $base_k$, exp_k und $result_k$ die Werte der Variablen nach dem k-ten Schleifendurchlauf)
Tipp: Unterscheiden Sie in Ihrem Induktionsbeweis die beiden Fälle, ob exp_k gerade ist oder nicht!

b) Folgern Sie daraus, dass das Programm mit der gewünschten Nachbedingung stoppt.

Aufgabe 2)

Zeigen Sie, dass das folgende Programm DIV und MOD ausrechnet, indem Sie eine geeignete Invariantenbedingung für die Schleife formulieren und mit vollständiger Induktion beweisen und daraus die am Ende angegebene Nachbedingung folgern:

```
(x ≥ 0) ∧ (y > 0) ∧ x,y ∈ ℤ          φ

div := 0;
mod := x;
while mod ≥ y do
   begin
      mod := mod - y;
      div := div + 1;
   end

(x = div * y + mod) ∧ (0 ≤ mod < y)   ψ
```

Tipp: Die Invariantenbedingung ist ein Teil der Nachbedingung.

Aufgabe 3)

Was berechnet die folgende Prozedur? Was ist die schwächste Vorbedingung dafür? (Beweis!)

Hinweis: Beweis durch vollständige Induktion über einen der Parameter.

```
procedure f(m,n: integer): integer

begin
   if (m <= 0)
     return 1
   else
     return m * f(m-1, n);
end;
```

Ersetzen Sie die Bedingung für den nichtrekursiven Aufruf durch (m < 0).
Was wird dann berechnet?

Aufgabe 4)

Gegeben sei folgende Funktion f:

```
procedure f(x, y, z: ℕ): ℕ;
begin
   if (x ≤ y) then
      return z
   else
     return f(x-1, y, z+1);
end;
```

a) Berechnen Sie f (10,7,3)

b) Was berechnet f(x,y,z) im allgemeinen? Beweisen Sie das durch vollständige Induktion über einen der Parameter!

Aufgabe 5)

Betrachten Sie folgende Funktion:

```
function f(x : R+) : R;
begin
   if x<=1 then
      return x²
   else
      return f(x/10);
end;
```

a) Beschreiben Sie in Worten, was diese Funktion berechnet.

b) Begründen Sie, warum Sie die Aussage von a) nicht mit vollständiger Induktion beweisen können.

c) Versuchen Sie dennoch, eine nachvollziehbare Begründung zu geben, warum die Aussage von a) gilt.

Aufgabe 6)

Betrachten Sie folgende Funktion:

```
function f(x : ℕ) : ℕ;
begin
   if x = 1 then
      return x
   else
      if x MOD 2 = 1
         then return f(x/2)
         else return f(3x+1)
end;
```

a) Beschreiben Sie und begründen Sie, was die Funktion berechnet, wenn sie zu einem Ende kommt.

b) Warum kann man nicht wie in Aufgabe 5 begründen, dass die Funktion immer zu einem Ende kommt?

c) Kann man aus b) zwingend folgern, dass es Fälle gibt, in denen die Funktion nicht zu einem Ende kommt?
Tipp: Googeln Sie mal nach Ulam-Collatz!