Bodo Eichstädt / Stephanie Wilke


# 2. Aufgabe

Geben Sie, umgangssprachlich, einen Algorithmus an, der überprüft, ob eine gegebene Zahl n eine Primzahl ist oder nicht. Als Elementaroperationen seien Vergleiche und Addition, Substraktion, Division und Multiplikation zwischen ganzen Zahlen erlaubt. Die Division soll dabei den ganzzahligen Anteil und den Rest liefern. Wieviele solcher Operationen führt Ihr Algorithmus in Abhängigkeit von der Zahl n aus ?

# Beispiellösung:

Wir wollen prüfen, ob n prim ist.

1. Wenn n == 2 ist, dann gebe aus dass 2 eine Primzahl ist und breche ab.
2. Wenn n gerade ist (n mod 2 == 0), dann gebe aus dass n keine Primzahl ist und breche ab.
3. Setze Register z auf 3.
4. Wenn n ohne Rest durch z teilbar ist (n mod z == 0), dann gebe aus, dass n keine Primzahl ist und breche ab.
5. Setze z auf z+2.
6. Wenn z kleiner n ist dann gehe zu Schritt 3.
7. Ausgabe: “n ist Prim”.

**Begründung der Korrekheit:**
Der Algorithmus funktioniert in der dargestellten Form, indem man zuerst ausschließt, dass die Zahl n keine gerade Zahl ist. Gerade Zahlen (ausser der 2) sind nie Primzahlen. (Schritt 1 und 2) Weiterhin lassen sich alle nicht-Primzahlen in Primteiler zerlegen, die stets kleiner sind als n. Die 1 und n scheiden als Primteiler aus, da jede Zahl durch 1 und sich selbst teilbar ist. Ebenso scheidet die 2 aus, da wir das ja zuvor (Schritt 2) mit dem Test auf eine gerade Zahl geprüft haben.
Beginnend mit der Zahl 3 überprüfen wir in Zweierschritten alle Zahlen kleiner n, ob sie Primteiler von n sind. (Schritt 4 bis 6)

**Effizienz:**

| Schritt Nr. | Anzahl der Operationen | Häufigkeit der Ausführung |
|---|---|---|
| 1 | 1 | 1 |
| 2 | 2 | 1 |
| 3 | 1 | 1 |
| 4 | 2 | (n-3)/2 |
| 5 | 1 | (n-3)/2 |
| 6 | 1 | (n-3)/2 |

Das macht in der Summe:
1*1+2*1+1*1+(2+1+1)*((n-3)/2)= 4+2*(n-3)= 2n-2
Für eine Zahl n werden also 2n-2 Operationen ausgeführt, um zu prüfen ob sie prim ist. Nicht als Operation wurde hier die Ausgabe eingerechnet.