

Primzahltests 3 und 4

1. Erstellen Sie eine Prozedur *isprime3*, die mit dem folgenden Primzahltest entscheidet, ob eine gegebene Zahl *n* Primzahl ist.

Test 3: Alle Zahlen kleiner als 2 sind keine Primzahlen.
 Gilt für alle $a \in \mathbb{N}$ ($1 < a < n$) mit $\text{ggT}(a, n) = 1$: $a^{n-1} \equiv 1 \pmod{n} \Rightarrow n$ ist Primzahl

Hinweise: Eingabe: Zahl *n* Ausgabe: *n* Primzahl \Rightarrow **TRUE**; *n* keine Primzahl \Rightarrow **FALSE**

- Fragen:** 1. Welche der Zahlen 10, 17, 307, 341, 561, 1105, 1729 werden als Primzahlen erkannt?

2. Bei welchen der unter 1. genannten Zahlen versagt der Test? Wie heißen die Zahlen?

2. Erstellen Sie eine Prozedur *isprime4*, die mit dem folgenden Primzahltest entscheidet, ob eine gegebene Zahl *n* Primzahl ist.

Test 4: Alle Zahlen kleiner als 2 sind keine Primzahlen.
 Die Zahlen 2 und 3 und 5 sind Primzahlen.
 Gilt für $a = 2, 3$ und 5 : $\text{ggT}(a, n) = 1$ und $a^{n-1} \equiv 1 \pmod{n} \Rightarrow n$ ist Primzahl

- Fragen:** 1. Welche der Zahlen 10, 17, 307, 341, 561, 1105, 1729 werden als Primzahlen erkannt?

2. Bei welchen der unter 1. genannten Zahlen versagt der Test? Wie heißen die Zahlen?

3. Vergleichen Sie die Prozeduren *isprime*, *isprime1*, *isprime2*, *isprime3* und *isprime4* bezüglich der benötigten **Rechenzeit** bei Überprüfung der gegebenen Primzahlen und tragen Sie die Messwerte in die Tabelle ein. Nutzen Sie `startzeit:=time(): isprime(?); (time()-startzeit)*ms`; **Messen Sie die Rechenzeit eines Tests bei den folgenden größeren Zahlen nicht mehr, sobald die Rechenzeit des Tests einmal 1000 ms überschritten hat.**

Rechenzeiten der Primzahltests in Millisekunden

Primzahl	isprime	isprime1	isprime2	isprime3	isprime4	
1009						
10007						
100003						
100000000003						
$2^{600} + 187$						

Woran liegt es, dass die einzelnen Tests so viel Zeit benötigen?

isprime1: _____

isprime2: _____

isprime3: _____

isprime4: _____

Verändern Sie *isprime4* in *isprime4powermod*, indem Sie für die Berechnung der Reste statt z.B. $2^{(n-1)} \pmod{n}$ den Befehl `powermod(2, n-1, n)` benutzen. Messen Sie die Rechenzeiten bei Überprüfung der Primzahlen für die neue Prozedur und tragen Sie die Messwerte in die Tabelle ein.

4. Vergleichen Sie die Prozeduren *isprime*, *isprime1*, *isprime2*, *isprime3* und *isprime4* bezüglich der **Sicherheit** ihrer Testaussagen. Geben Sie für jeden Primzahltest an, was das Ergebnis **TRUE** und was das Ergebnis **FALSE** bei Eingabe einer natürlichen Zahl größer als 1 jeweils bedeutet.

Sicherheit der Primzahltests

Test	TRUE	FALSE
<i>isprime</i>		
<i>isprime1</i>		
<i>isprime2</i>		
<i>isprime3</i>		
<i>isprime4</i>		

Zusatz

1. Erstellen Sie eine Prozedur *isprime4Liste*, die alle mit *isprime4powermod* erkannten Primzahlen in eine Liste schreibt.

Hinweise: Eingabe: Zahl n Ausgabe: Liste mit von *isprime4powermod* erkannten Primzahlen von 1 bis n

- Fragen:** 1. Wie viele Zahlen werden im Intervall von 1 bis 5000 als Primzahlen erkannt? _____
 2. Welche der erkannten Zahlen von 1 bis 5000 sind keine Primzahlen?

3. Überprüfen Sie Ihre Aussage bei 2.: Erstellen Sie eine Primzahlliste von 1 bis 5000 mit dem *isprime*- Befehl von MuPAD. Vergleichen Sie die Anzahl der Primzahlen. Welche Zahlen wurden bei *isprime4powermod* fälschlicherweise für Primzahlen gehalten?

2. Erstellen Sie eine Prozedur *isprime5*, die mit dem folgenden Primzahltest entscheidet, ob eine gegebene Zahl n Primzahl ist.

Test 5: Alle Zahlen kleiner als 2 sind keine Primzahlen.
 Die Zahlen 2, 3 und 5 sind Primzahlen.

$$\text{Gilt für } a = 2, 3, 5: \text{ggT}(a, n) = 1 \text{ und } a^{\frac{n-1}{2}} \equiv \pm 1 \pmod{n} \Rightarrow n \text{ ist Primzahl}$$

Hinweise: Der Operator *powermod* von MuPAD berechnet jeweils die positiven Reste.

- Fragen:** 1. Welche der Zahlen 10, 17, 307, 341, 561, 1105, 1729, 2821 werden als Primzahlen erkannt?

2. Bei welchen der unter 1. aufgeführten Zahlen versagt der Test?

3. Vergleichen Sie *isprime5* bezüglich der Sicherheit und der Rechenzeit mit den anderen Primzahltests. Was ist der Vorteil dieses Tests gegenüber *isprime4*?

4. Erstellen Sie mit *isprime5* eine Prozedur *isprime5Liste*. Welche Zahlen bestimmt *isprime5* auf den ersten 15000 Zahlen falsch? Wie viele Zahlen bestimmt *isprime5* auf den ersten 20000 (50000) Zahlen falsch?
