Lektion 3 : Beispielprogramme | **Anwendung : Primzahltest**

In dieser Anwendung von Lektion 3 verwenden Sie die in den vorherigen Lektionen erworbenen Konzepte, um Algorithmen zu programmieren, mit denen Sie Ihr Wissen über Zahlen und insbesondere Primzahlen verfeinern können.

Lernziele :

- Anwendung von Tests und Schleifen in einem Python-Programm über **Primzahlen**

Eine **Primzahl** hat als Teiler nur die 1 und sich selbst.
Zum Beispiel :

- 1 ist keine Primzahl, da sie nur einen Teiler (sich selbst) hat.
- 7 ist eine Primzahl mit den Teilern 1 und 7.
- 8 ist keine Primzahl, denn sie hat die 4 Teiler 2,4,8 und 1

Die ersten Primzahlen sind : 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, …
Es gibt unendlich viele Primzahlen

Mit einem Programm soll die 2020. Primzahl bestimmt werden. Es handelt sich dabei nicht um eine Funktion wie bisher, da zum Start kein Funktionsname aufgerufen werden muss. Der Hauptteil des Programmes wird ohne Einrückung unter allen Funktionen angefügt.

Das Programm

- Legen Sie ein neues Programm « **primz** » an.
- Das Modul « **math** » muss vorab geladen werden.
- Das Programm enthält eine Funktion **prim(n)**, die den Test auf Primzahl durchführt. Die zu testende Zahl **n** wird dabei schrittweise euklidisch von **2** bis $\lfloor\sqrt{n}+1\rfloor$ durch **k** dividiert.
- Ist **r = 0 (keine Primzahl)**, so ist **prim(n) = 0**, andernfalls **1**.
- Im Hauptteil des Programmes werden die zu testenden Zahlen **N** erzeugt, und die Anzahl der Primzahlen **np** gezählt, bis die **Grenze 2020** erreicht ist.
- Es dauert dann eine Weile, bis das Ergebnis dargestellt wird.
- Die Rechenzeit lässt sich verkürzen, indem man die geraden Zahlen bei der Primzahlbestimmung überspringt.
- Um ein Ergebnis zu sehen, muss eine **print()** – Anweisung eingefügt werden.