**Hochschule Worms**
University of Applied Sciences
***Fachbereich Informatik***
Prof. Dr. Bernd Ruhland

# Objektorientierte Programmierung
Übungsaufgabe

## Aufgabenstellung:
## Überladen der Operatoren Autoinkrement / Autodekrement sowie des Subskript-Operators

Bitte lesen Sie den Text genau durch und halten Sie sich an die Vorgaben.

Erstellen Sie eine Klasse ***`cPrimZahl`*** mit dem Attribut ***aktprim***.

Eine Instanz dieser Klasse soll **eine** Primzahl (Erklärung siehe unten) zwischen 2 und ***`maxprim`*** aufnehmen (also im Attribut primzahl speichern).

Legen Sie ***`maxprim`*** mittels einer globalen Konstanten auf 10000 fest.

Der Konstruktor soll als Parameter eine natürliche Zahl (positive Ganzzahl) im Bereich zwischen 1 und ***`maxprim`*** erhalten (Vorgabewert 1), das ist die Startzahl.

Zu dieser Startzahl soll der Konstruktor bereits die nächst höhere oder gleiche Primzahl ermitteln und diese als Attributwert speichern. Ist die Startzahl also bereits eine Primzahl, so ist diese bereits der gefundene Wert. Ist die nächst höhere Primzahl größer als ***`maxprim`***, so soll die höchste Primzahl gefunden werden, die kleiner ist als ***`maxprim.`***

*Überladen Sie den Ausgabeoperator* **<<** *für die Klasse* ***`cPrimZahl`***. Es soll der Wert der Primzahl in den Ausgabestrom gegeben werden.

Überladen Sie für die Klasse ***`cPrimZahl`*** die Operatoren **++** (Autoinkrement) und **--** (Autodekrement), in der **Präfix**-Version (also linksseitig) als Methoden (Mitgliedsfunktionen).

Der Inkrement-Operator soll auf die nächst größere Primzahl springen. Wenn er über ***`maxprim`*** hinauskommt, soll er auf der größten Primzahl stehen bleiben, die kleiner als ***`maxprim`*** ist.

Der Dekrement-Operator soll auf die nächst kleinere Primzahl springen. Wenn er unter die 2 kommt, soll er auf der 2 stehen bleiben.

Überladen Sie für die Klasse ***`cPrimZahl`*** den Subskript-Operator **[ ]** so, dass er als Wert die dem Index nächste (gleiche oder höhere) Primzahl findet, speichert und zurückgibt. Ist der Indexwert bereits eine Primzahl, so ist dieser bereits der Wert. Ansonsten wird die nächst höhere Primzahl gesucht.

Beispiel:

```
cPrimZahl pz; int i;
i = pz[18];  // findet und speichert den Wert 19
i = pz[19];  // speichert den Wert 19, der Indexwert ist bereits eine Primzahl
i = pz[20];  // findet und speichert den Wert 23
```

Wird der Subskript-Operator mit einem Indexwert < = 0 aufgerufen, so gibt er den Wert zurück, den die Primzahl gerade hat. Der Wert der Primzahl bleibt unverändert.

Schreiben Sie ein Hauptprogramm, in welchem Sie in einer Schleife die Eingaben ‚+' und ‚-, sowie ‚s' und ‚e' zulassen.

Wenn ein ‚+' eingegeben wird, verwenden Sie den Autoinkrement-Operator.

Wenn ein ‚-, eingegeben wird, verwenden Sie den Autodekrement-Operator.

Zeigen Sie nach jeder Operation den Wert des Attributs mit dem Ausgabeoperator an.

Wenn ein ‚s' eingegeben wird, soll zudem eine Zahl eingegeben werden; mit dieser Zahl machen Sie die Subskription, z.B. **`i = pz[s]`** und geben Sie den Rückgabewert **`i`** aus.

Wenn ein ‚e' eingegeben wird, beendet sich das Programm.

Sie können wenn Sie möchten auch ein eigenes Hauptprogramm mit Ihren Ideen schreiben oder das Programm s.o. erweitern.

---

Tipp: Sie können Primzahlen mittels des Modulo-Verfahrens bestimmen. Dazu ist folgende einfache Hilfsfunktion geeignet. Sie können diese auch als private Hilfsmethode implementieren:

```
bool isPrime(int n) {            // Pruefung einer Zahl auf Primzahl
   for(int i=2; i<=n/2; i++) {
         if (n % i) {
               continue;
         }
         else {
               return false;
         }
    }
    return true;
}
```