Programmierkurs I (Inf., W-Info.), Übungsblatt 9

Claus/Weicker, Wintersemester 03/04 Abgabetermin: 08.01.2004, 23:59 Uhr

Bitte beachten Sie die Hinweise von Aufgabenblatt 8 sowie die Angaben in den jeweiligen Spezifikationsdateien $*$.ads.

Aufgabe 1: Primzahlen (mittel) 5+7(+2) Punkte

Primzahlen sind die natürlichen Zahlen größer als 1, die nur durch 1 und sich selbst teilbar sind. Die erste Primzahl ist 2, die zweite ist 3, die 25. Primzahl ist 97 usw.

a) Auf der Webseite der Veranstaltung ist die Datei `primzahlen.ads` erhältlich, welche die Schnittstellen für zwei Funktionen definiert. Implementieren Sie diese Funktionen in einer Datei `primzahlen.adb` und geben diese Datei ab.

- Die Funktion

  ```
  function Berechne_Primzahl (Nummer: in Natural) return Natural;
  ```

  berechnet für die Eingabe `Nummer`$= n$ die n-te Primzahl und gibt diese als Funktionswert zurück.

- Die Funktion

  ```
  function Berechne_Index (Abstand: in Natural) return Natural;
  ```

  berechnet die kleinste Nummer i derjenigen Primzahl, für die als erstes die Distanz zur nächsten Primzahl größer als der Aufrufparameter `Abstand` ist, d.h. es muss $p_{i+1} - p_i >$`Abstand` gelten, wobei p_i die i-te Primzahl bezeichne. Zum Beispiel wäre für einen Aufruf mit `Abstand`$= 3$ das Ergebnis 4, da

$$\begin{aligned} p_2 - p_1 &= 3 - 2 = 1 \leq 3 \\ p_3 - p_2 &= 5 - 3 = 2 \leq 3 \\ p_4 - p_3 &= 7 - 5 = 2 \leq 3 \\ p_5 - p_4 &= 11 - 7 = 4 > 3 \end{aligned}$$

  Hinweis: Um die Aufgabe zu lösen, dürfen Sie ein Array mit fester Größe von maximal 2^{16} Einträgen verwenden, um bereits gefundene Primzahlen zu speichern.

b) **Zusatzaufgabe:** Geben Sie Ihr Testprogramm ab, mit dem Sie Ihre Abgabe für die erste Teilaufgabe auf Korrektheit geprüft haben. Erläutern Sie insbesondere die unterschiedlichen Testfälle in Ihrem Testprogramm.

Aufgabe 2: Von drauß' vom Walde ... (mittel bis schwer) 8(+2) Punkte

Der Weihnachtsmann hat sich beim Verteilen der Geschenke in den berüchtigten Matrixwald verirrt. In diesem Wald stehen die Bäume stets im gleichen Abstand in Reih' und Glied. Wenn wir annehmen, dass der Standpunkt des Weihnachtsmanns mit der Koordinate $(0, 0)$ bezeichnet ist, so steht an jeder anderen Koordinate (x, y), wobei x und y ganze Zahlen sind, ein Baum.
Implementieren Sie die in der Datei `weihnachtsmann.ads` vorgegebene Funktion

```
function Ist_Sichtbar(X, Y: in Integer) return Boolean;
```

die überprüft, ob ein Weihnachtsbaum an der angegebenen Position für den Weihnachtsmann sichtbar ist. Die Funktion gibt den Wert `false` zurück, falls die Sicht auf den Baum durch einen anderen Baum verdeckt wird. Der Einfachheit halber dürfen Sie annehmen, dass die Bäume (und der Weihnachtsmann selbst) punktförmig sind.

Beispiel: Für die Werte $-4 \leq X, Y \leq 4$ können Sie die Sichtbarkeit der folgenden Grafik entnehmen. Nur die mit einem `*` gekennzeichneten Bäume sind für den Weihnachtsmann `W` sichtbar:

	−4	−3	−2	−1	0	1	2	3	4
4		*		*		*		*	
3	*		*	*		*	*		*
2		*		*		*		*	
1	*	*	*	*	*	*	*	*	*
0				*	W	*			
−1	*	*	*	*	*	*	*	*	*
−2		*		*		*		*	
−3	*		*	*		*	*		*
−4		*		*		*		*	

a) Geben Sie die Implementation in der Datei `weihanchtsmann.adb` ab.

b) **Zusatzaufgabe:** Geben Sie Ihr Testprogramm ab, mit dem Sie Ihre Abgabe für die erste Teilaufgabe auf Korrektheit geprüft haben. Erläutern Sie insbesondere die unterschiedlichen Testfälle in Ihrem Testprogramm.

Hinweis: Falls Sie für die Lösung dieser Aufgabe den Abstand zwischen zwei Koordinaten (x_1, y_1) und (x_2, y_2) benötigen, können Sie diesen mit der üblichen euklidischen Distanz

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}.$$

Die Funktion kann jedoch auch anders implementiert werden. berechnen.

Hinweise

- Zur Abgabe wird das eClaus-System verwendet:
  `http://eclaus.informatik.uni-stuttgart.de`
- Die Abgabe zu jeder Teilaufgabe besteht aus einem kompilierbaren Ada-Quelldatei. In jeder Aufgabe wird ein Dateiname vorgegeben. Verwenden Sie diesen bitte für das Hauptprogramm. Auch sind in der Aufgabe Angaben zu Ein- und Ausgabetexten sowie zur Formatierung der Ausgabe enthalten. Bitte folgen Sie diesen Angaben so genau wie möglich.
- Beachten Sie beim Programmieren bitte die folgenden Hinweise („kleine Programmierrichtlinie“).
  - Halten Sie einzelne Bestandteile überschaubar, z.B. indem Sie nur eine Anweisung pro Zeile schreiben, sowie pro Zeile höchstens 80 Zeichen, höchstens 40 Zeilen pro Prozedur/Funktion, nicht mehr als 800 Zeilen pro Datei und höchstens 5 Parameter bei Prozeduren/Funktionen benutzen.
  - Bezeichner sollen selbsterklärend und maximal 15 Zeichen lang sein. Bezeichner enthalten nur Buchstaben, den Bindestrich oder den Unterstrich.
  - Durch Einrückung um 2 Zeichen soll die logische Gliederung eines Programms verdeutlicht werden. Beispielsweise wird der Anweisungsteil einer IF-Verzweigung eingerückt, während „`end if;`“ nicht mehr eingerückt wird. Auch auf der nächsten Zeile fortgesetzte Zeilen werden um 2 Zeichen eingerückt.
  - Zu Beginn des Programms muss in Kommentaren das Konzept und die Lösungsidee des Programms ausführlich erläutert werden.
  - Auch im Programmtext sind Kommentare einzufügen, um den Code zu erläutern.
- Beachten Sie, dass jede Abgabe individuell vom jeweiligen Studierenden erstellt werden muss. Werden von den Tutoren Plagiate erkannt, d.h. exakte oder leicht modifizierte Kopien, werden für die Aufgabe keine Punkte vergeben. Falls Sie die Lösung der Aufgaben vor der Abgabe in Gruppen besprechen, achten Sie darauf, dort nur das generelle Konzept abzuklären und die Programmierung jedem selbst zu überlassen.

- Bei Fragen wenden Sie sich bitte an Ihren Tutor oder an die Übungsleitung: `Karsten.Weicker@fmi.uni-stuttgart.de` oder Tel. 7816-337
- Weitere Veranstaltungshinweise einschließlich der Übungsblätter finden Sie unter: `http://www.informatik.uni-stuttgart.de/ifi/fk/lehre/ws03-04/ada95/`