ETH
Eidgenössische Technische Hochschule Zürich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo

computer graphics lab

Departement Informatik
Dr. R. Peikert - Christian Sigg

7. November 2003

37-847 Informatik Übung 3 WS03/04

1) Sortieren durch Einfügen

Für eine beliebige Konstante `N` deklarieren Sie den Array

```
int a[N];
```

und initialisieren ihn mit Zufallszahlen (s.u.). Schreiben Sie nun ein Programm, das diesen Array sortiert. Gehen Sie dabei nach folgendem Muster vor:

```
for(i=0; i<n; ++i) {
    /* sortiere den Array a[0], ..., a[i] */
}
```

Offenbar ist vor dem `i`-ten Schritt der Array `a[0], ..., a[i-1]` bereits sortiert. Um den Array `a[0], ..., a[i]` zu sortieren, muss also lediglich das Element `a[i]` an der richtigen Stelle eingefügt werden.

Hinweis: Im Headerfile `stdlib.h` ist der Zufallszahlengenerator `rand()` definiert. Ein Aufruf von `rand()` erzeugt eine zufällige ganze Zahl zwischen `0` und $2^{15}-1$. Damit sich die Resultate für verschiedene Programmausführungen unterscheiden, muss der Generator einmal mit einer „zufälligen" ganzen Zahl `i` initialisiert werden. Dies geschieht mit `srand(i)`. Die Zahl können Sie mit `std::cin` einlesen oder mit der Anweisung `(unsigned)time(NULL)` erzeugen. Letzteres gibt die Aktuelle Systemzeit in Sekunden zurück und ist im Headerfile `time.h` definiert.

2) Primzahlen

Eine natürliche Zahl *x* ist eine Primzahl, wenn *x* nur durch sich selbst und durch 1 teilbar ist. Eratosthenes hat folgende Methode zur Berechnung der Primzahlen vorgeschlagen. Schreibe zunächst alle Zahlen der Reihe nach auf:

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, ...

Beginne dann von links alle Vielfachen abzustreichen, zuerst die von 2:

1, 2, 3, 5, 7, 9, 11, ...

dann alle Vielfachen von 3, usw. Die nicht durchgestichenen Zahlen sind dann offensichtlich die gesuchten Primzahlen.

Benutzen Sie das oben dargestellte Verfahren (Sieb des Eratosthenes) zur Berechnung aller Primzahlen im Bereich $n = 1, ..., 1000$.

Abgabetermin: 18. November 2003