WBS TRAINING

Programmierung(1)

UNSER ♥ SCHLÄGT FÜR BILDUNG

Agenda

- **Zufallszahlen**
  - Motivation
  - Darstellung in PAP, Struktogramm, Pseudocode
  - Syntax in ANSI C
    - vorgegebener Zahlenbereich
    - selbstbestimmter Zahlenbereich
    - Initialisierung des Zufallsgenerators
- Ausführliches Training + Ergebnisbesprechung
- Fachpraktische Anwendungen

Zufallszahlen – Motivation

- Programme werden (in der Regel) erst interessant, wenn diese **interaktiv** auf (z.B.) User-Eingaben reagieren (und also nicht bei jedem Programmdurchlauf zu den selben Ergebnissen führen).
  - Während der Testphase solcher Programme, muss der Programmierer jedoch **selbst dafür sorgen**, dass diese (User)-Eingaben vollständig durchgeführt werden, was im Einzelfall recht zeitaufwendig sein kann.
- Eine Abhilfe bieten hier **Zufallszahlen**, die vom Programm selbst erzeugt werden und somit die entsprechenden (User)-Eingaben simulieren können.

Hinweise:

- Auf das Auslosen von einzelnen Zeichen werden wir im Moment noch verzichten, und darauf erst zurückkommen, wenn wir uns näher mit der Darstellung der Character im **ASCII-Code** beschäftigt haben.
- Dies gilt erst Recht für das Auslosen von Wörtern oder Sätzen (Zeichenketten), für die uns noch einige Voraussetzungen fehlen.
- Technisch betrachtet werden wir zunächst nur in die Lage gesetzt, **ganze Zahlen** auszulosen. Eine entsprechende Erweiterung auf Kommazahlen wird Gegenstand einer Übungsaufgabe sein.

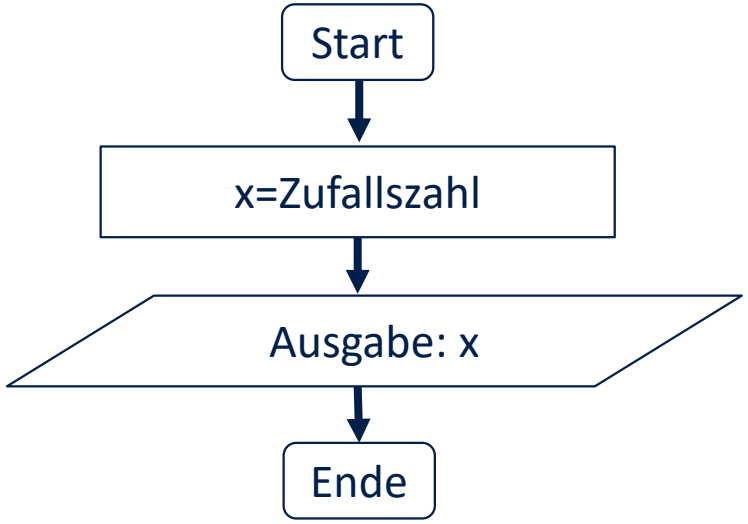
Zufallszahlen – Beispielaufgabe

- Als einen ersten Einstieg betrachten wir zunächst eine bewusst einfach gehaltene Aufgabe.
- Dies erscheint sinnvoll, da wir uns zum einen auf die eigentliche Thematik konzentrieren wollen, und da zum anderen weder bei PAP, Struktogramm noch Pseudocode eine festgelegte Symbolik für Zufallszahlen existiert.
- Ferner werden wir bei der Erstellung des entsprechenden Quellcodes einen lehrreichen Einstieg in dieses Themenfeld erhalten und daher im Anschluss leicht einsehen können, dass die von ANSI C zur Verfügung gestellte Technologie im allgemeinen noch ergänzt werden muss.

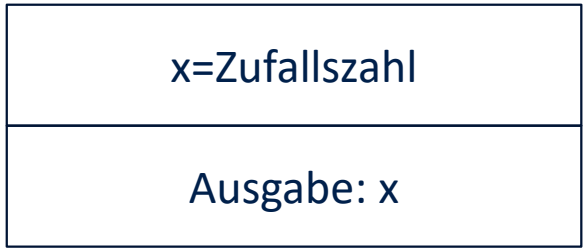
Aufgabenstellung:

Das Programm soll zunächst eine Variable **x** mit einer beliebigen (ganzzahligen) Zufallszahl initialisieren.
Anschließend soll der Wert von x auf der Konsole ausgegeben und das Programm beendet werden.

Beispielaufgabe – PAP



Beispielaufgabe – Struktogramm



Beispielaufgabe – Pseudocode

```
Programm „Zufalls-Beispiel“
{
        x=Zufallszahl
        Ausgabe: x
}
```

Beispielaufgabe – Quellcode

```
# include<stdio.h>
# include<stdlib.h>

main()
{
        int x;

        x= rand();
        printf("Die Variable x hat aktuell den Wert: %d",x);
}
```

```
Die Variable x hat aktuell den Wert: 41
```

Quellcode – Notwendige Ergänzungen(1a)

- Ein mehrfaches Ausführen des bisherigen Quellcodes wird nun aber zeigen, dass **stets der selbe** „Zufalls“-Wert ausgegeben wird. Dies kann wie folgt erklärt werden:
  - (Klassische) Computer sind nicht dazu ausgelegt, zufällige Aktionen auszuführen. Vielmehr erwarten wir von ihnen, dass die von uns aufgerufenen Befehle **verlässlich** (und also *nicht* zufällig) abgearbeitet werden.
  - Um dennoch mit (Pseudo)-Zufallszahlen arbeiten zu können, wird eine **künstlich erzeugte Liste** von „wild durcheinander gewürfelten“ Zahlen verwendet, die von der bereits vorgestellten rand()-Funktion der Reihe nach ausgelesen werden.
    - Wenn daher **innerhalb eines einzigen Programmablaufs** durch wiederholtes Aufrufen von rand() mehrere (Pseudo)-Zufallszahlen ermittelt werden, so wirkt diese Zahlenfolge auf den ersten Blick tatsächlich zufällig.
    - Bei einem **erneuten Aufruf des selben Programmes** wird sich dann aber zeigen, dass die **selbe Zahlenfolge** erscheint, da schlicht die identische Liste in der selben Reihenfolge ausgelesen wird.
    - Um dieses Problem zu lösen, werden wir die Startposition stets **in Abhängigkeit** der aktuell gültigen „**Uhrzeit**“ ermitteln. *(genauer: Anzahl der Sekunden seit dem 01.01.1970)*
      - Eine solche Zuweisung der Startposition („**Initialisierung des Zufallsgenerators**“) ist dann zwar streng genommen keineswegs zufällig, bleibt für den User aber dennoch vollkommen unvorhersehbar, zumal er die zigtausende Einträge umfassende Liste von (Pseudo)-Zufallszahlen üblicherweise nicht auswendig lernte … oder Monk heißt ;-)

Notwendige Ergänzungen(1a) – Initialisierung des Zufallsgenerators

```
# include<stdio.h>
# include<stdlib.h>
# include<time.h>


main()
{
        int x;

        srand(time(NULL));
        x= rand();
        printf("Die Variable x hat aktuell den Wert: %d",x);
}
```

Quellcode – Notwendige Ergänzungen(1b)

- Obwohl die Initialisierung des Startgenerators im Prinzip bereits ausreichen sollte, zeigt sich bei zwei kurz nacheinander stattfindenden Programm-Durchläufen jedoch, dass der **erste** ermittelte Zufallswert eines Programm-Durchlaufs nur unwesentlich größer ist, als der erste Zufallswert des vorangegangen Durchlaufs.
- Dies kann dadurch umgangen werden, dass nach der Initialisierung des Zufallsgenerators der erste Aufruf der rand()-Funktion bewusst ungenutzt bleibt. (siehe Musterlösung der kommenden Folie)

Wichtiger Hinweis:

Die Initialisierung des Startgenerators darf pro Programm **nur einmalig** stattfinden (und sollte vor dem ersten Aufruf der rand()-Funktion geschehen), da sich in allen anderen Fällen die ermittelten „Zufallswerte“ in der identischen Reihenfolge wiederholen.

Notwendige Ergänzungen(1b) – einmalig ungenutzter rand()-Aufruf

```
# include<stdio.h>
# include<stdlib.h>
# include<time.h>


main()
{
        int x;

        srand(time(NULL));
        rand();
        x= rand();
        printf("Die Variable x hat aktuell den Wert: %d",x);
}
```

Quellcode – Notwendige Ergänzungen(2a)

- Wir haben bisher (zumindest dem Anschein nach) „beliebige“ Zufallszahlen ermitteln lassen.
- Tatsächlich lagen diese Werte aber zwischen (beiderseits einschließlich) 0 und einem gesetzten „Maximalwert“, den wir mittels der (Integer)-Konstanten RAND_MAX auslesen können:

```
# include<stdio.h>
# include<stdlib.h>

main()
{
        printf(“Der Maximalwert der Zufallsliste lautet: %d“,RAND_MAX);
}
```

```
Der Maximalwert der Zufallsliste lautet: 32767
```

Quellcode – Notwendige Ergänzungen(2b)

- Oft wollen wir allerdings nur eine bestimmte Folge von Zufallszahlen zulassen, Beispiele:
  - Würfeln: ……. Zahl zwischen (beiderseits einschließlich) 1 und 6
  - Lotto: ……. Zahl zwischen (beiderseits einschließlich) 1 und 49
  - Allgemein: ….. Zahl zwischen (beiderseits einschließlich) a und b
- Diese Einschränkung auf einen Bereich von (unmittelbar aufeinander folgenden, ganzzahligen) Werten gelingt uns aber relativ leicht mittels des Modulo-Operators:
  - Würfeln: …… rand()%6+1
  - Lotto: ……. rand()%49+1
  - Allgemein: … rand()%(b-a+1)+a

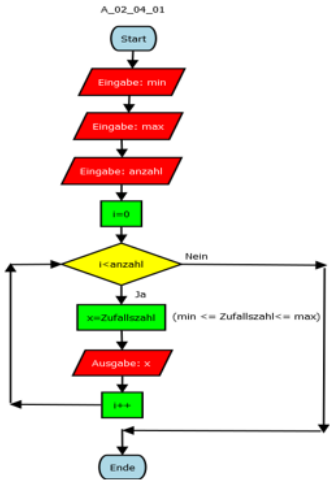
Hinweis:
Gelegentlich könnte es auch von Interesse sein, einen beliebigen Wert aus einer Menge von *nicht* unmittelbar aufeinanderfolgenden Werten auszulosen. Also zum Beispiel einen Wert aus der Menge {-13; 5; 77; …}. Um dieses Problem elegant zu lösen, benötigen wir noch Kenntnisse, die wir uns erst im weiteren Verlauf des Kurses aneignen werden. Die Frage, wie dies (weniger elegant) bereits mit unseren aktuellen Kenntnissen gelingt, wird Gegenstand einer Übungsaufgabe sein.

Zufallswerte – Gemeinsame Übung A_02_04_01

WBS TRAINING

Aufgabe_02_04_01

Gegeben sei der folgende PAP:



Aufgabenstellung:
Bitte erstellen Sie dazu einen geeigneten **Quellcode** in ANSI C.

WBS TRAINING AG
Lorenzweg 5
D-12099 Berlin
Amtsgericht Berlin HRB 68531
Sitz der Gesellschaft: Berlin

Vorstand:
Heinrich Kronbichler,
Joachim Giese
Aufsichtsrat (Vorsitz): Dr. Daniel Stadler
USt-IdNr.: DE 209 768 248

GLS Gemeinschaftsbank eG
IBAN: DE18 4306 0967 1146 1814 00
BIC: GENODEM1GLS

WBS TRAINING

VIELEN DANK FÜR IHRE AUFMERKSAMKEIT!