Dr. Johannes Mayer
Axel Blumenstock
Christian Ehrhardt
Abteilung Angewandte Informationsverarbeitung

8. November 2005
Blatt 3

Allgemeine Informatik III / Systemnahe Software I (WS 2005/2006)

Abgabetermin: 15. November 2005

Magic Seven (10 Punkte)

Da Sie für Ihre Altersvorsorge (siehe Blatt 1) ja nun regelmäßige Einnahmen brauchen, die Bildverarbeitung (Blatt 2) sich aber doch als nicht so lukrativ erwiesen hat, versuchen Sie es nun als Anbieter von Glücksspielen. Vielleicht sind die Leute in unsicheren Zeiten ja etwas risikofreudiger?

Unter `http://www.mathematik.uni-ulm.de/users/martin/bots/m7/` haben Sie ein vielversprechendes (oder: viel versprechendes?) Spiel namens *Magic Seven* gefunden, das ähnlich wie Roulette funktioniert. Gedreht wird an einem Rad, das eine gedächtnislose und gleichverteilte Folge von Zufallszahlen im Bereich von 1 bis 7 erzeugt. Wurde eine 7 gezogen, wird ein zweites Mal gedreht. Der Spieler kann auf eine von 17 Kategorien setzen, nämlich auf die sieben einzelnen Zahlen, sechs Paare und vier Tripel.

Tabelle 1 gibt an, das wievielfache seines Einsatzes er zurückerhält, gegeben seine Wette und das Ergebnis des Zufallsexperiments. Setzt der Spieler also beispielsweise auf „yellow“, so bekommt er bei den Ausgängen 2, 5, 7+2 oder 7+5 jeweils das Dreifache seines Einsatzes ausbezahlt, sonst nichts.

Ziel dieser Aufgabe ist, ein Programm zu erstellen, das *einem* Benutzer erlaubt, *Magic Seven* zu spielen. Von einem vorgegebenen Startkapital soll er einen beliebigen ganzzahligen Betrag auf *eine* Kategorie setzen können, woraufhin der Rechner das Spiel ausführt und den Kontostand aktualisiert. Eine komplette Spielsession könnte also wie folgt aussehen:

```
Ihr Kontostand betraegt: 100 Euro.
Wieviel moechten Sie einsetzen? Alles
Bitte geben Sie eine nichtnegative Ganzzahl ein.
Wieviel moechten Sie einsetzen? 200
Sie duerfen nicht mehr einsetzen, als Ihr Konto hergibt (100 Euro).
Wieviel moechten Sie einsetzen? 10
Bitte setzen Sie auf ein Feld (1-7/s/m/b/g/y/r/l/u/o/e): q
Diese Wette ist nicht moeglich.
Bitte setzen Sie auf ein Feld (1-7/s/m/b/g/y/r/l/u/o/e): e
```

Wette \ Zufallsrad	1	2	3	4	5	6	7	7	7	7	7	7	7
							1	2	3	4	5	6	7
1	6						6						
2		6						6					
3			6						6				
4				6						6			
5					6						6		
6						6						6	
7 (magic)							6	6	6	6	6	6	12
small	3	3					3	3					
medium			3	3					3	3			
big					3	3					3	3	
green	3			3			3			3			
yellow		3			3			3			3		
red			3			3			3			3	
lower	2	2	2				2	2	2				
upper				2	2	2				2	2	2	
odd	2		2		2		2		2		2		
even		2		2		2		2		2		2	

Tabelle 1: Die Gewinntabelle

```
Das Rad bleibt auf der 4 stehen.
Sie haben eben 20 Euro gewonnen.
Ihr Kontostand betraegt: 110 Euro.
Wieviel moechten Sie einsetzen? 20
Bitte setzen Sie auf ein Feld (1-7/s/m/b/g/y/r/l/u/o/e): 7
Das Rad bleibt auf der 6 stehen.
Sie haben leider nichts gewonnen.
Ihr Kontostand betraegt: 90 Euro.
Wieviel moechten Sie einsetzen? 20
Bitte setzen Sie auf ein Feld (1-7/s/m/b/g/y/r/l/u/o/e): 7
Das Rad bleibt auf der 7 stehen.
Beim zweiten Lauf bleibt es auf der 3 stehen.
Sie haben eben 120 Euro gewonnen.
Ihr Kontostand betraegt: 190 Euro.
Wieviel moechten Sie einsetzen? (Vom Benutzer beendet)
```

Die Implementierungsdetails seien ganz Ihnen überlassen. Ihr Programm sollte aber auf jeden Fall

- robust auf Fehleingaben reagieren,
- den Lauf des Rades sowie Gewinne/Verluste ausgeben,
- mit Control-D (das das Ende des Eingabestroms markiert) beendet werden können.

Die nachfolgenden Schritte sind nur als ein möglicher, sinnvoller Weg zu verstehen.

(a) Die Wette des Benutzers soll mit einem einzelnen Zeichen ('1'–'7' bzw. die jeweiligen Anfangsbuchstaben der Kategorien) angegeben werden können. Schreiben Sie dazu eine Funktion

```
int inputchar(char *prompt)
```

welche die gegebene Eingabeaufforderung `prompt` auf der Standardausgabe ausgibt[1], ein Zeichen vom Benutzer einliest und als Funktionsergebnis zurückliefert. Zuvor sollen sämtliche weiteren Zeichen bis einschließlich des abschließenden newline (`'\n'`) aus der Eingabe gelesen und ignoriert werden.

Beachten Sie dabei, dass die Funktion *getchar()* auch die Konstante `EOF` zurückgeben kann, wenn der Benutzer zur Beendigung des Eingabestroms Control-D drückt. Sobald ein `EOF` gelesen wurde, soll kein weiterer Aufruf von *getchar()* mehr erfolgen, sondern die Funktion *inputchar()* mit einem geeigneten Rückgabewert abgebrochen werden.

Testen Sie Ihre Funktion mit einer geeigneten *main()*-Routine, bis sie zu Ihrer Zufriedenheit arbeitet.

(b) Implementieren Sie unter Verwendung der Funktion *rand()*[2] eine einfache Routine, die (pseudo-) zufällig eine Ganzzahl zwischen 1 und 7 (einschließlich) liefert.

(c) Damit wiederum erstellen Sie nun eine Funktion

```
int gain(char bet)
```

welche für die gegebene Wette `bet` des Benutzers (eines der zuvor eingelesenen Zeichen '1'–'7', 's', 'm', 'b' usw.) das Spiel durchführt und den Gewinnfaktor ausgibt, wie Sie ihn obiger Tabelle entnehmen können.

(d) Mit einem zunächst noch fest einprogrammierten Spieleinsatz können Sie nun die Hauptroutine des Spiels schreiben, die die Wette des Benutzers erfragt, das Spiel ausführt und den Kontostand aktualisiert.

(e) Für die Eingabe des Spieleinsatzes stellen Sie eine weitere Funktion

```
int inputint(char *prompt)
```

bereit, welche eine Ganzzahl einliest, in ähnlicher Weise wie *inputchar* den Rest der Zeile ignoriert und erst dann zurückkehrt, wenn aus der Eingabe tatsächlich ein gültiger Wert gelesen werden konnte, der zudem nicht negativ ist. Für ein gelesenes EOF soll -1 zurückgegeben werden.

Beachten Sie, dass *scanf()* einen nützlichen Rückgabewert liefert.

(f) Machen Sie Ihr Programm schließlich noch robust gegenüber Fehleingaben wie falsche Zeichen für die Wette oder zu große Spieleinsätze.

[1] `char *prompt` war noch nicht Gegenstand der Vorlesung, aber Sie können diese Funktionsdeklaration einfach abtippen und danach die Variable `prompt` in Funktionen wie *fputs()* verwenden wie die gewohnten Zeichenkettenkonstanten `"..."`. Auch beim Aufruf der Funktion übergeben Sie eine Zeichenkettenkonstante: `inputchar("Gib ein Zeichen: ");`

[2] Siehe `man 3 rand`. Gegebenenfalls müssen Sie die Header-Datei `stdlib.h` einbinden. Damit nicht bei jedem Programmlauf dieselbe Folge an Zufallsziffern erzeugt wird, empfiehlt sich eine einmalige Initialisierung mit `srand(time(0))`. Hierin liefert *time(0)* die aktuelle Zeit in Sekunden seit 1970. Für diese Funktion muss `time.h` eingebunden werden.

Zusatzaufgabe (4 Punkte) Erweitern Sie Ihr Programm, so dass es vor einem Lauf des Rades mehrere Einsätze erlaubt (bis das Geld eben ausgeht). Eine derartige Session könnte wie folgt aussehen:

```
Ihr Kontostand betraegt: 100 Euro.
Moechten Sie (s)etzen oder s(p)ielen? s
Bitte setzen Sie auf ein Feld (1-7/s/m/b/g/y/r/l/u/o/e): m
Wieviel moechten Sie einsetzen? 20
Sie wetten auf "m" 20 Euro.
Ihr Kontostand betraegt: 80 Euro.
Moechten Sie (s)etzen oder s(p)ielen? s
Bitte setzen Sie auf ein Feld (1-7/s/m/b/g/y/r/l/u/o/e): 7
Wieviel moechten Sie einsetzen? 60
Sie wetten auf "7" 60 Euro.
Sie wetten auf "m" 20 Euro.
Ihr Kontostand betraegt: 20 Euro.
Moechten Sie (s)etzen oder s(p)ielen? p
Das Rad bleibt auf der 7 stehen.
Beim zweiten Lauf bleibt es auf der 3 stehen.
Ihr Einsatz von 60 Euro auf "7" erzielt einen Gewinn von 360 Euro
Ihr Einsatz von 20 Euro auf "m" erzielt einen Gewinn von 60 Euro
Insgesamt haben Sie 420 Euro gewonnen.
Ihr Kontostand betraegt: 440 Euro.
Moechten Sie (s)etzen oder s(p)ielen? (Vom Benutzer beendet)
```

Viel Erfolg!