

## 10 Quest

### 10.1 Aufgabe

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

static int gotit(float* sum, int max, int picked[], float roots[])
{
    int i;
    float help = 1.0;
    for (i = 0; i < max; i++)
        if (picked[i] != 0) help *= roots[i];
    *sum += help;
}

static void pick(int index, int count, int max,
                int picked[], float roots[], float* sum)
{
    if (count == 0)
        gotit(sum, max, picked, roots);
    else
    {
        int i;
        for (i = index; i <= max-count; i++)
        {
            picked[i] = 1;
            pick(i+1, count-1, max, picked, roots, sum);
            picked[i] = 0;
        }
    }
}

static void vieta(int nr, float roots[], float coeffs[])
{
    int i, picked[nr];
    float sum;
    for (i = 0; i < nr; i++)
        picked[i] = 0;
    for (i = 1; i <= nr; i++)
    {
        sum = 0;
        pick(0, i, nr, picked, roots, &sum);
        if (i % 2 == 0) coeffs[nr-i] = sum;
        else coeffs[nr-i] = -sum;
    }
}
```

```
int main(int argc, char** argv)
{
    int nr;
    printf("Wie_viele_Nullstellen_soll_das_Polynom_besitzen?_");
    scanf("%d", &nr);
    float roots[nr], coeffs[nr];
    int i;
    for (i = 0; i < nr; i++)
    {
        printf("Bitte_geben_Sie_die_%d.te_Nullstelle_ein:_", i+1);
        scanf("%f", &roots[i]);
    }

    vieta(nr, roots, coeffs);
    printf("Das_folgende_Polynom_hat_die_eingegebenen_Nullstellen:\n x^%d+_+",
           nr);
    for (i = nr-1; i > 0; i--)
    {
        printf("%f_x^%d+_+", coeffs[i], i);
    }
    printf("%f\n", coeffs[0]);
}
```

### 10.2 Aufgabe

Es sei  $n$  die Anzahl der virtuellen Seiten und  $C_i$  die Häufigkeit des Auftretens der Distanzzeichenkette  $i$ . Für  $m$  Seiten Seiten wird die Seitenfehlerrate  $F_m := \sum_{k=m+1}^n C_k + C_\infty$  definiert. Für Aufgabe 2 aus Übung 9 ergibt sich daraus:

$i$	1	2	3	4	5	6	7	$\infty$
$C_i$	0	1	1	3	5	6	1	8
$F_i$	20	19	18	15	10	9	8	8

Folglich beträgt die minimale Seitenfehlerrate 8.

### 10.3 Aufgabe

Durch DMA ist es angeschlossenen Geräten möglich ohne CPU mit dem Arbeitsspeicher zu kommunizieren. Dies entlastet sowohl den Prozessor, als auch sorgt es für schnellere Übertragungsraten. Es werden insbesondere drei grundlegende Übertragungskonzepte unterschieden:

1. Single Bus Transfer (Cycle Stealing)  
Hier wird nach Übertragen genau eines Datums die Kontrolle vom DMA-Controller wieder an die CPU übergeben.
2. Burst Mode (Block Transfer)  
Hier behält der DMA-Controller die Kontrolle solange, bis ein ganzer Block übertragen wurde.

## 3. Demand Transfer Mode

Hier werden so viele Zyklen ausgeführt, wie angefordert wurden (unabhängig von der Blockgröße).

Probleme durch Framgentierung können durch die folgenden Verfahren gelöst werden: TODO

1. try-contiguous
2. dma-chain
3. scatter-gather

**10.4 Aufgabe**

Das Spoolen soll verhindern, dass ein Drucker von einem Prozess blockiert wird. In einem Mehrprozesssystem

sendet also ein Prozess seine Druckaufträge an einen Dämon, der diese, sobald sie komplett empfangen wurden, an den Drucker schickt.

**10.5 Aufgabe**

$$7200 \text{ U/min} \Rightarrow 120 \text{ U/sec} \Rightarrow 1 \text{ U} \hat{=} \frac{1000}{120} \text{ ms}$$

$$200 \text{ Sektoren} \Rightarrow \frac{1000}{200 \cdot 120} = \frac{1}{24} \text{ ms pro Sektor}$$

$$24 \cdot \frac{1}{24} \text{ ms} = 1 \text{ ms} \hat{=} \text{Suchzeit} \Rightarrow 24 \text{ Sektoren Zylindersatz benötigt}$$

**10.6 Aufgabe**

Die Antwort hängt von den Daten der Festplatte ab.

**10.7 Aufgabe**

Methode	Summanden	Summe
FCFS	10, 12, 2, 18, 38, 34, 32	146 $\Rightarrow$ 876 ms
SSF	0, 2, 12, 4, 4, 36, 2	60 $\Rightarrow$ 360 ms
Aufzug UP	0, 2, 16, 2, 30, 4, 4	58 $\Rightarrow$ 348 ms
Aufzug DOWN	0, 10, 4, 4, 20, 16, 2	56 $\Rightarrow$ 342 ms

**10.8 Aufgabe**

$$(20 \cdot 100 \cdot 3) \text{ Bytes/sec} = 6000 \text{ Bytes/sec}$$