

Name, Vorname:

Hilfsmittel:

Keine

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

AUFGABEN

1)

12P

Betrachten Sie den folgenden Programmausschnitt:

```
...
int main(){
    int zahl = 4;
    int quad = 8;
    —————> 1
    quadrat(zahl, &quad);
    —————> 2
    printf("%d hoch 2 = %d\n", zahl, quad);
    return 0;
}

void quadrat(int z, int *zq){
    *zq = z * z;
}
```

Durch die Deklaration der Variablen zahl und quad werden die folgenden Zellen

	Adresse	Inhalt
zahl	08151	?
quad	04711	?

im Arbeitsspeicher reserviert.

- a) Welchen Wert hat der Inhalt der Variablen zahl und quad an der Stelle 1 im Programm ?
- b) Welchen Wert hat z und zq beim Aufruf von quadrat(zahl, &quad) ?
- c) Was bewirkt die Anweisung *zq = z * z an welcher Adresse im Arbeitsspeicher ?
(konkreten Wert der Adresse und deren Inhalt angeben!)
- d) Welchen Wert hat der Inhalt der Variablen zahl und quad an der Stelle 2 im Programm ?

2)

13P

Sie wollen ein Programm schreiben, das abhängig von der Eingabe entweder die Summe oder die Differenz oder das Produkt oder den Quotienten zweier Zahlen liefert.

Sie wollen dazu die Funktion `tr` (wie Taschenrechner) benutzen, die Sie aber wegen Arbeitsüberlastung von einem "Programmierknecht" implementieren (programmieren) lassen. Entwerfen Sie für den "Programmierknecht" eine **Beschreibung** (Leistungsbeschreibung) dieser Funktion mit dem im Unterricht verwendeten Schema. (Kein Programm !!!)

3)

12P

Betrachten Sie den folgenden Programmausschnitt:

```
...
int main() {
    float r;
    float u;
    r = 2;
    u = 3;
    → 1
    berechne_umfang (r, u);
    → 2
    printf("Radius= %f, Umfang= %f", r, u);
}

void berechne_umfang(float radius, float umfang) {
    umfang = 2 * 3.14 * radius ;
}
```

Durch die Deklaration der Variablen `r` und `u` werden die folgenden Zellen

	Adresse	Inhalt
<code>r</code>	0120	?
<code>u</code>	0130	?

im Arbeitsspeicher reserviert.

a) Welchen Wert hat der Inhalt der Variablen `r` und `u` an der Stelle 1 im Programm ?

b) Welchen Wert hat `radius` und `umfang` beim Aufruf von `berechne_umfang (r, u)` ?

c) Was bewirkt die folgende Anweisung im Arbeitsspeicher an der Adresse 0130 ?

```
umfang = 2 * 3.14 * radius;
```

d) Welchen Wert hat der Inhalt der Variablen `r` und `u` an der Stelle 2 im Programm ?

4)

13P

Programmieren Sie anhand der folgenden Beschreibung die dazugehörige Funktion:

```

/*****
/**
/**  int ersatz(double r1, double r2, int mod, double *rg)  **/
/**
/**
/**
/*#*****/
/*

```

Parameter:

```

(i) double r1>0:    erster Widerstandswert
(i) double r2>0:    zweiter Widerstandswert
(i) int mod:         10: Parallelschaltung
                    20: Reihenschaltung
(o) double *rg:     Gesamtwiderstand

```

Return:

```

(o) 0: Parallelschaltung oder Reihenschaltung wurde
    berechnet (mod ist 10 oder 20)
    -1: mod ist weder 10 noch 20

```

Beschreibung:

Berechnet in Abhängigkeit vom Modus mod (10 bedeutet eine Parallelschaltung, 20 bedeutet eine Reihenschaltung), den Widerstandswerten r1 und r2 den Ersatzwiderstand (Gesamtwiderstand) rg der Widerstandsschaltung.

*/

Lösungen

- 1) 12P
a) zahl: 4, quad: 8 1P + 1P
b) z: 4, zq: 04711 1P + 2P
c) In den Inhalt der Adresse 04711 wird der Wert 16 geschrieben. 4P
d) zahl: 4, quad: 16 1P + 2P

2) 13P
/*****/
/** **/
/** double tr (double z1, double z2, int mod) **/
/** **/
/*#*****/

Parameter:

- (i) double z1: erste Zahl
- (i) double z2!=0, wenn mod=4: zweite Zahl
- (i) int mode{1;2;3;4}:
 - 1: berechnet Summe z1+z2
 - 2: berechnet Differenz z1-z2
 - 3: berechnet Produkt z1*z2
 - 4: berechnet Quotient z1/z2

Return:

- (o) Ergebnis der gewünschten Operation (Summe, oder Differenz oder Produkt oder Quotient).

Beschreibung:

Berechnet in Abhängigkeit vom Modus mod:

Summe z1+z2, wenn mod = 1

Differenz z1-z2, wenn mod = 2

Produkt z1*z2, wenn mod = 3

Quotient z1/z2, wenn mod = 4

*/

Jede fehlende Zusicherung: -2 P

z2!=0 ist keine richtige Zusicherung, da dann z.B. auch 3 * 0 verboten würde: -2 P

Bemerkung zum Begriff Zusicherung:

Die Angabe beim Parameter z1:

z2!=0, wenn mod=4

und die Angabe:

mode{1;2;3;4}

nennt man Zusicherung. Das bedeutet, daß unter diesen Voraussetzungen der Programmierer dieser Funktion für die Korrektheit der Berechnungen dieser Funktion **garantiert**.

Je weniger Zusicherungen der Programmierer macht, desto größer wird der programmtechnische Aufwand für ihn, desto mehr "Intelligenz" muss er in die Funktion packen.

- 3) 12P
- a) $r = 2, u = 3$ 1P + 1P
- b) radius = 2, umfang = 3 1P + 2P
- c) nichts 4P
- d) $r = 2, u = 3$ 3P

4) 13P

```
int ersatz(double r1, double r2, int mod, double *rg){
    int r;
    if(mod==10){
        *rg=1/(1/r1+1/r2);
        r=0;
    }
    else if(mod==20){
        *rg=r1+r2;
        r=0;
    }
    else
        r=-1;
    return(r);
}
```

Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

AUFGABEN

Bemerkungen:

Bitte EVA-Prinzip genau beachten !!

1) Das folgende Programm soll die Anzahl (wenn alle drei Zahlen verschieden sind, ist die Anzahl gleich Eins) gleicher Zahlen von drei über Tastatur eingegebenen ganzen Zahlen berechnen und auf dem Bildschirm ausgeben.

a) Berichtigen Sie nur die **syntaktischen** Fehler, so dass das C-Programm lauffähig wird.

b) Geben Sie 3 **konkrete** Werte für z1, z2 und z3 an, für die das Programm eine falsche Ausgabe erzeugt. Es wird vorausgesetzt, dass der Benutzer auch ganze Zahlen eingibt.

```
woid main()
{
    int z1, z2, z3, anzahl,
    printf("1. Zahl eingeben\n");
    scanf("%d",&z1)
    fflush(stdin);
    printf("2. Zahl eingeben\n");
    scanf("%d",&z2);
    fflush(stdin);
    printf("3. Zahl eingeben\n");
    scanf("%d",&z3);
    fflush(stdin);

    if((z1==z2)&&(z2==z3))
        anzahl = 3;
    if(z1==z2)
        anzahl = 2;
    if(z1==z3)
    {
        anzahl = 2;
    }
    if(z2==z3)
    {
        anzahl = 2;
    }
    if((z1!=z2)&&(z2!=z3))
    {
        anzahl = 1;
    }
    printf("Anzahl gleicher Zahlen=\"%d\n", anzahl);
}
```

2) Das folgende syntaktisch korrekte Programm soll die Anzahl bestimmen, wie oft die Zahl 1 in zwei über Tastatur eingegebenen, ganzen Zahlen vorkommt und diese Anzahl dann auf dem Bildschirm ausgeben. Es wird vorausgesetzt, dass der Benutzer auch ganze Zahlen eingibt.

a) Was macht das Programm im Endeffekt in Wirklichkeit ?

b) Warum ?

c) Vereinfachen Sie das Programm so weit wie möglich, so daß es genau das gleiche macht, wie das bei a) real gegebene Programm.

```
#include "stdafx.h"
#include "stdio.h"

void main()
{
    int z1, z2, anz;
    printf("1. Zahl eingeben\n");
    scanf("%d", &z1);
    fflush(stdin);
    printf("2. Zahl eingeben\n");
    scanf("%d", &z2);
    fflush(stdin);
    anz = 0;
    if(z1=1)
        anz++;
    if(z2=1)
        anz++;
    printf("Anzahl der Einsen = %d\n", anz);
}
```

3) Von einer Schreinerei kommt folgender Auftrag:

Schreiben Sie ein C-Programm, das den Umfang und die Fläche eines Rechtecks berechnet, dessen Seiten (Länge, Breite) über Tastatur eingegeben werden.

Das Programm soll an den 2 über Tastatur eingegebenen Zahlen erkennen, ob es sich überhaupt um 2 positive Zahlen handelt bzw. ob es sich um ein Quadrat oder um ein Rechteck handelt und davon abhängig eine entsprechende Ausgabe auf den Bildschirm bringen.

Bem:

Zuerst ein Struktogramm des zugehörigen Programms machen.

4) Erstellen Sie ein Struktogramm zu dem Programm, das den Ersatzwiderstand von **maximal** 3 parallel geschalteten Widerständen berechnet. Die Widerstände (Einheit: Ohm) müssen über Tastatur eingegeben werden. Wenn für einen Widerstand ein Wert kleiner oder gleich 0 eingegeben wird, wird der Ersatzwiderstand der bisher eingegebenen Widerstände berechnet und ausgegeben (bzw. eine entsprechende Meldung, wenn kein Ersatzwiderstand existiert).

Danach wird das Programm **beendet** (ohne auf eine weitere Eingabe zu warten).

D.h. es kann kein Ersatzwiderstand bzw. der Ersatzwiderstand von 1, 2 oder 3 parallelen Widerständen berechnet werden.

Lösungen:

Allgemeines

1) Punkteverteilung (50 P = 1,0)

1a)	5P		2a)	4P		3)	13P		4)	13P		
1b)	7P		2b)	4P								
			2c)	4P								

2) Abzüge

Struktogramm stimmt nicht mit Programm überein: -3P

Kein EVA-Prinzip: -3P

Logischer Fehler: -3P

3) Bemerkungen

Bei 1a) waren syntaktische - keine sematische - Fehler verlangt.

--> `woid main` `void` falsch geschrieben

--> `int z1, z2, z3, anzahl,` Semikolon statt Komma am Schluss

--> `scanf("%d",&z1)` Semikolon fehlt am Schluss

--> `scanf("%d, &z3");` Anführungszeichen nach "%d", nicht nach z3

--> `printf("Anzahl gleicher Zahlen = \"%d\\n\", anzahl);` Anführungszeichen vor %d

Aufgabe 1:

b)

`z1 = z2 = z3 = 100`

Aufgabe 2:

a) `anz` bekommt den Wert 2 zugewiesen

b) Weil in den Bedingungen statt `==` das Zeichen `=` verwendet wurde.

Damit wird die Bedingung `z1 = 1` bzw. `z2 = 2` immer wahr.

c)

```
void main()  
{  
    printf("Anzahl der Einsen ist 2");  
}
```

Übungsaufgabe für die Ferien

Es soll ein Taschenrechner (er soll die 4 Grundrechenarten beherrschen) programmiert werden.

a/A: Berechnet die Summe

b/B: Berechnet die Differenz

c/C: Berechnet das Produkt

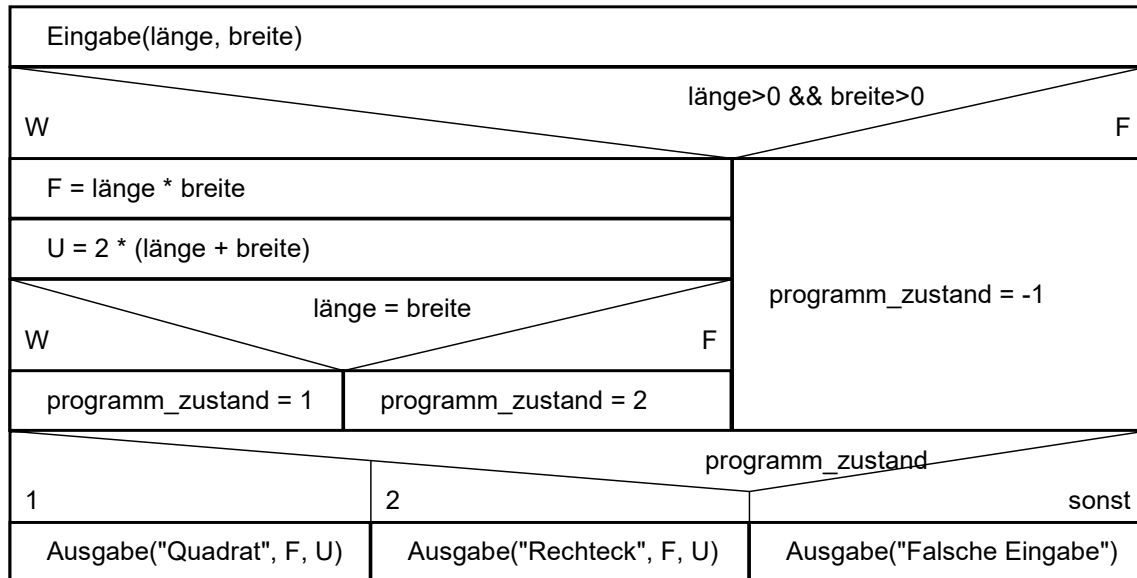
d/D: Berechnet den Quotient

sonst: Programmende

Dieser soll auf 2 Arten (der Anwender kann dies auswählen) betrieben werden können: als Erwachsenentaschenrechner oder als Kindertaschenrechner. Der Kindertaschenrechner soll bei Eingabe einer negativen Zahl bzw. eines negativen Ergebnisses sofort eine entsprechende Meldung bringen und dann das Programm sofort beenden.

Außerdem soll der Taschenrechner (egal ob als Erwachsenentaschenrechner oder als Kindertaschenrechner genutzt) bei Division durch Null sofort eine entsprechende Meldung bringen und dann das Programm sofort (ohne Rechnung) beenden.

Aufgabe 3



Bem:

Bei Aufgabe 4 habe ich (vielleicht schafft es jemand von Ihnen) es leider nicht geschafft, mit dem Programm Struktured in den Kopf der Switch-Anweisung die Fälle -11, -12, -13 einzutragen. Deshalb habe ich dies in der dann nachfolgenden Zeile gemacht.

Aufgabe 4

Eingabe(R1)			
W		R1 > 0	
		F	
Eingabe(R2)		programm_zustand = -11	
W			
R2 > 0			
F			
Eingabe(R3)		programm_zustand = -12	
W			
R3 > 0			
		F	
programm_zustand = 0		programm_zustand = -13	
programm_zustand			
1	2	3	sonst
Fall -11: Rg = -1	Fall -12: Rg = R1	Fall: -13: Rg = 1/(1/R1+1/R2)	Rg = 1/(1/R1+1/R2+1/R3)
programm_zustand == -11			
W		F	
Ausgabe("Ungültige Eingabe")		Ausgabe("Ersatzwiderstand=", Rg)	

KLAUSUR 1 Programmierpraktikum 2BK11 2.12.2002 Zeit: 90 Minuten

Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

Bemerkungen:

Bitte EVA-Prinzip genau beachten !!

1) Schreiben Sie folgendes C-Programm (Kleiner Taschenrechner):

Zuerst müssen zwei Fließkommazahlen eingegeben werden.

Dann wird in Abhängigkeit von der Eingabe des Anwenders eine Rechenoperation (Summe, Differenz, Produkt, Quotient) durchgeführt und auf dem Bildschirm ausgegeben.

- a/A: Berechnet die Summe
- b/B: Berechnet die Differenz
- c/C: Berechnet das Produkt
- d/D: Berechnet den Quotient
- sonst: Programmende

2) Verändern Sie das Programm so, daß bei Division durch Null das Programm (ohne Rechnung) beendet und eine entsprechende Meldung auf dem Bildschirm ausgegeben wird.

3) Verändern Sie das Programm wie folgt:

Zuerst erscheint die folgende Maske:

- a/A: Berechnet die Summe
- b/B: Berechnet die Differenz
- c/C: Berechnet das Produkt
- d/D: Berechnet den Quotient
- sonst: Programmende

Wenn vom Anwender nicht das Programmende ausgewählt wird, muß er zwei Zahlen eingeben. Dann wird in Abhängigkeit von der Eingabe des Anwenders eine Rechenoperation (Summe, Differenz, Produkt, Quotient) durchgeführt und auf dem Bildschirm ausgegeben. Bei Division durch Null soll das Programm (ohne Rechnung) allerdings beendet und eine entsprechende Meldung auf dem Bildschirm ausgegeben werden.

4) Optional soll das Programm bei 3) noch einen Kindertaschenrechner enthalten.

Am Anfang des Programms soll gefragt werden, ob man es als Kindertaschenrechner oder als Erwachsenentaschenrechner benutzen will.

Als Erwachsenentaschenrechner soll es wie 3) funktionieren.

Als Kindertaschenrechner sollen bei Verwendung von negativen Zahlen (als Eingaben oder als Ergebnis) und bei Division durch Null das Programm (ohne Rechnung) beendet und eine entsprechende Meldung auf dem Bildschirm ausgegeben werden.

KLAUSUR 1 Programmierpraktikum 2BK11 9.12.2002 Zeit: 90 Minuten

Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

Bemerkungen:

Bitte EVA-Prinzip genau beachten !!

1) Schreiben Sie das folgende C-Programm, das den Brutto- bzw. Nettopreis berechnet:
Zuerst muss ein Geldbetrag in Euro eingegeben werden. Dann muß ein Prozentsatz eingegeben werden.

Dann wird in Abhängigkeit von der Eingabe des Anwenders der Brutto- bzw. Nettopreis davon berechnet und auf dem Bildschirm ausgegeben.

b/B: Berechnet den Bruttopreis

n/N: Berechnet den Nettopreis

sonst: Programmende

2) Verändern Sie das Programm so, daß bei Eingabe eines negativen Geldbetrages bzw. eines negativen Prozentsatzes das Programm sofort (ohne etwas zu rechnen, bzw. ohne auf eine weitere Eingabe zu warten) beendet und eine entsprechende Meldung auf dem Bildschirm ausgegeben wird.

3) Verändern Sie das Programm wie folgt:

Zuerst erscheint die folgende Maske:

b/B: Berechnet den Bruttopreis

n/N: Berechnet den Nettopreis

sonst: Programmende

Wenn vom Anwender nicht das Programmende ausgewählt wird, muß er einen Geldbetrag und einen Prozentsatz eingeben. Dann wird in Abhängigkeit von der Eingabe des Anwenders der Bruttopreis bzw. der Nettopreis berechnet und auf dem Bildschirm ausgegeben.

Bei Eingabe eines negativen Geldbetrages bzw. eines negativen Prozentsatzes muss das Programm sofort (ohne etwas zu rechnen, bzw. ohne auf eine weitere Eingabe zu warten) beendet und eine entsprechende Meldung auf dem Bildschirm ausgegeben werden.

4) Optional soll das Programm bei 3) zusätzlich noch die Möglichkeit bieten, den Zins und das Gesamtkapital zu berechnen, das man für einen bestimmten Zinssatz von einer Bank für ein eingesetztes Kapital erhält.

Am Anfang des Programms soll gefragt werden, ob man es für eine Netto/Bruttoberechnung oder für eine Zins/Kapitalberechnung verwenden will.

Falls man für den Zinssatz und das eingesetzte Bankkkapital eine negative Zahl eingibt, soll das Programm sofort (ohne etwas zu rechnen, bzw. ohne auf eine weitere Eingabe zu warten) beendet und eine entsprechende Meldung auf dem Bildschirm ausgegeben werden.

KLAUSUR 2 Programmierpraktikum 2BK11 9.12.2002 Zeit: 45 Minuten
Gruppe A Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

Bemerkungen:

Bitte EVA-Prinzip so weit - wie möglich - einhalten !!

(Dies ist in der folgenden Aufgabe, mit dem bisher im Unterricht Erlernten, nicht 100 % möglich).

1) Schreiben Sie ein C-Programm, das den Ersatzwiderstand einer Parallelschaltung (in Ohm) von Widerständen berechnet.

Der Anwender muß erst die Anzahl der Widerstände und dann die einzelnen Werte dieser Widerstände über Tastatur eingeben.

Beachten Sie, daß der Anwender (z.B. ein Hacker) alle möglichen ganzzahligen Werte über Tastatur eingeben kann. Das Programm soll dann darauf entsprechend reagieren.

KLAUSUR 2 Programmierpraktikum 2BKI1 9.12.2002 Zeit: 45 Minuten
Gruppe B *Name, Vorname:*

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
 - NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
 - Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
 - Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

Bemerkungen:

Bitte EVA-Prinzip so weit - wie möglich - einhalten !!

(Dies ist in der folgenden Aufgabe, mit dem bisher im Unterricht Erlernten, nicht 100 % möglich).

1) Schreiben Sie ein C-Programm, das den Mittelwert von Noten berechnet.

Der Anwender muß zuerst die Anzahl der Noten und dann die einzelnen Noten über Tastatur eingeben.

Beachten Sie, daß der Anwender (z.B. ein Hacker) alle möglichen ganzzahligen Werte über Tastatur eingeben kann. Das Programm soll dann darauf entsprechend reagieren.

Name, Vorname:

Hilfsmittel:

Keine

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

AUFGABEN

1) Schreiben Sie ein C-Programm, das die Fakultät einer ganzzahligen Zahl (≥ 1) berechnet.

$$n! = n * (n-1) * (n-2) * \dots * 1$$

Beispiele: $1! = 1$ $2! = 2 * 1$ $3! = 3 * 2 * 1$ $4! = 4 * 3 * 2 * 1$

Sie geben über Tastatur eine ganzzahlige Zahl (≥ 1) ein. Von dieser Zahl soll die Fakultät berechnet werden und das Ergebnis auf dem Bildschirm ausgegeben werden.

Beispiel:

Sie geben z.B. die Zahl 4 ein. Auf dem Bildschirm wird dann ausgegeben:

$$n! = 24$$

Bemerkung:

Es soll berücksichtigt werden, dass der Anwender alle möglichen ganzzahligen Werte eingeben kann. Das Programm soll darauf entsprechend reagieren.

KLAUSUR 2 Programmierpraktikum 2BKI1 Nachtermin 1 Zeit: 45 Minuten

Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punktabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

1) Ein Anfangskapital wird zu einem Zinssatz eine bestimmte Anzahl Jahre auf einem Konto angelegt, ohne die Zinsen abzuheben.

Schreiben Sie eine Funktion, die das Endkapital berechnet.

Bemerkung:

Es soll berücksichtigt werden, dass der Anwender alle möglichen ganzzahligen Werte eingeben kann. Das Programm soll darauf entsprechend reagieren.

Mathematische Anleitung anhand eines Beispiels:

Voraussetzungen: $K_0 = 100$ DM; $p = 1$ %;

n : Anzahl der Jahre,

z_n : neu hinzukommende Zinsen

K_n : Kapital nach n Jahren

n	z_n	K_n
0	0	100
1	$100 \cdot 0,01 = 1$	$100 + 1 = 101$
2	$101 \cdot 0,01 = 1,01$	$101 + 1,01 = 102,01$
3	$102,01 \cdot 0,01 = 1,0201$	$102,01 + 1,0201 = 103,0301$

KLAUSUR 2 Programmiertheorie 2BKI1 Nachtermin 1 Zeit: 45 Minuten

Name, Vorname:

Hilfsmittel:

Keine

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

AUFGABEN

1) Ein Anfangskapital wird zu einem Zinssatz eine bestimmte Anzahl Jahre auf einem Konto angelegt, ohne die Zinsen abzuheben.

Schreiben Sie ein Programm, das das Endkapital berechnet.

Bemerkung:

Es soll berücksichtigt werden, dass der Anwender alle möglichen ganzzahligen Werte eingeben kann. Das Programm soll darauf entsprechend reagieren.

Mathematische Anleitung anhand eines Beispiels:

Voraussetzungen: $K_0 = 100$ DM; $p = 1$ %;

n : Anzahl der Jahre,

z_n : neu hinzukommende Zinsen

K_n : Kapital nach n Jahren

n	z_n	K_n
0	0	100
1	$100 \cdot 0,01 = 1$	$100 + 1 = 101$
2	$101 \cdot 0,01 = 1,01$	$101 + 1,01 = 102,01$
3	$102,01 \cdot 0,01 = 1,0201$	$102,01 + 1,0201 = 103,0301$

Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Die Lösungen müssen als ablauffähige C-Programme dargestellt werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Die rote Farbe darf nicht benutzt werden.
- Aufgabenblätter bitte auch abgeben.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Die Programme müssen ausgedruckt werden.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

AUFGABEN

1) Geben Sie die syntaktischen Fehler und die Fehler, die zur Laufzeit entstehen können, an.

```
#include "stdafx.h"
#include <stdio.h>

void main()
{
    int i = 3;
    int v[3]={10,9,15,8};
    double w[i];

    v[2] = 10;
    w[i+1]=v[2];
    printf("r=%d\n",v[i-3]);
}
```

2) Es müssen die Umsätze (größer oder gleich Null) einer Firma in ein Feld eingelesen werden.

Die Umsätze müssen über Tastatur eingegeben werden. Wenn für einen Umsatz ein Wert kleiner als Null eingegeben wird, wird die Eingabe beendet. Dann wird der Gesamtumsatz (die Summe dieser Umsätze) berechnet und auf dem Bildschirm ausgegeben.

Erstellen Sie dazu ein **Struktogramm**, kein C-Programm !!

Bemerkung:

- 1) EVA-Prinzip beachten
- 2) Es dürfen keine nicht reservierten Zellen eines Feldes überschrieben werden.

KLAUSUR 3 Programmierpraktikum 2BK11 07.04.2002 Zeit: 90 Minuten
Gruppe A Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
 - NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
 - Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
 - Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
 - Die Programme müssen ausgedruckt werden.
 - Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

1) Schreiben Sie ein C-Programm, das die Umsätze (größer oder gleich Null) einer Firma in ein Feld einliest.

Die Umsätze müssen über Tastatur eingegeben werden. Wenn für einen Umsatz ein Wert kleiner als Null eingegeben wird, wird die Eingabe beendet. Dann wird der Gesamtumsatz (die Summe dieser Umsätze) berechnet und auf dem Bildschirm ausgegeben.

Bemerkung:

- 1) EVA-Prinzip beachten
- 2) Es dürfen keine nicht reservierten Zellen eines Feldes überschrieben werden.

KLAUSUR 3 Programmierpraktikum 2BKI1 28.4.2003 Zeit: 45 Minuten
Gruppe B *Name, Vorname:*

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

1) Schreiben Sie ein C-Programm, in dem der Anwender eine Zeichenkette über Tastatur eingibt und in der ersten Zeile eines zweidimensionalen Feldes abspeichert. Dann soll das Programm diese eingegebene Zeichenkette in umgekehrter Reihenfolge in der zweiten Zeile des zweidimensionalen Feldes abspeichern und danach auf dem Bildschirm ausgeben.

Beispiel:

Eingabe: 2BKI1

Ausgabe: 1IKB2

Name, Vorname:

Hilfsmittel:

selbstgeschriebene DIN A4-Seite

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Die Lösungen müssen als ablauffähige C-Programme dargestellt werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Die rote Farbe darf nicht benutzt werden.
- Aufgabenblätter bitte auch abgeben.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programnteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

AUFGABEN

1) Betrachten Sie den folgenden Programmausschnitt:

```
...  
void main()  
{  
double kapital = 100;  
double zinssatz = 5;  
double zinsen = 60;  
double endkapital = 200;  
→ [1]  
endkapital=kapital_zins(kapital, zinssatz, &zinsen); [2] ←  
→ [3]  
}  
  
double kapital_zins(double k, double p, double *z)  
{  
double erg;  
*z = k*p/100;  
erg = k+k*p/100;  
return(erg);  
}
```

Durch die Deklaration der Variablen kapital, zinssatz, zinsen und endkapital werden die folgenden Zellen im Arbeitsspeicher reserviert.

	Adresse	Inhalt
kapital	0800	?
zinssatz	0808	?
zinsen	0816	?
endkapital	0824	?

a) Welchen Wert hat der Inhalt der Variablen kapital, zinssatz, zinsen und endkapital an der Stelle [1] im Programm ?

b) Welchen Wert hat k, p, und z gleich beim Beginn der Abarbeitung des Aufrufs an der Stelle [2]

c) Welchen Wert hat der Inhalt der Variablen kapital, zinssatz, zinsen und endkapital an der Stelle [3] im Programm ?

2) Sie wollen ein Programm schreiben, das **entweder** den arithmetischen Mittelwert $((a+b)/2)$ oder den harmonischen Mittelwert $(2ab/(a+b))$ zweier Zahlen liefert.

Sie wollen dazu eine Funktion benutzen, die Sie aber wegen Arbeitsüberlastung von einem "Programmierknecht" implementieren (programmieren) lassen.

Entwerfen Sie für den "Programmierknecht" eine **Beschreibung** (Leistungsbeschreibung) dieser Funktion mit dem im Unterricht verwendeten Schema. (Kein Programm !!!)

3) Betrachten Sie den folgenden Programmausschnitt:

```
...
void main()
{
    double grundseite, hoehe, flaeche;
    grundseite = 10;
    hoehe = 20;
    → 1
    berechne_flaeche(grundseite, hoehe, flaeche); 2 ←
    → 3
}

void berechne_flaeche(double g, double h, double f)
{
    f = g * h / 2 ;
}
```

Durch die Deklaration der Variablen grundseite, hoehe und flaeche werden die folgenden Zellen im Arbeitsspeicher reserviert.

	Adresse	Inhalt
grundseite	0700	?
hoehe	0708	?
flaeche	0716	?

a) Welchen Wert hat der Inhalt der Variablen grundseite, hoehe und flaeche an der Stelle 1 im Programm ?

b) Welchen Wert hat g, h und f gleich beim Beginn der Abarbeitung des Aufrufs an der Stelle 2

c) Welchen Wert hat der Inhalt der Variablen grundseite, hoehe und flaeche an der Stelle 3 im Programm ?

4) Programmieren Sie anhand der folgenden Beschreibung die dazugehörige Funktion:

```

/*****
**
** void minmax(int z1, int z2, char mod, int *erg)
**
**
**#*****
*/
Parameter:
    (i) int z1:      erste Zahl
    (i) int z2:      zweite Zahl
    (i) char mod:
                    'g': Maximum von z1 und z2
                    'k': Minimum von z1 und z2
    (o) int *erg:    Minimum oder Maximum (abhängig von "mod") von z1 und z2

Return:
    Kein

Beschreibung:
    Berechnet in Abhängigkeit vom Modus mod ('g' bedeutet das
    Maximum, 'k' bedeutet das Minimum) entweder das Minimum oder
    das Maximum der Zahlen z1 und z2
*/
```


KLAUSUR 4 Programmierpraktikum 2BK11 23.06.2003 Zeit: 60 Minuten

Gruppe A Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punktabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

Bemerkung:

1) Es muss genau ein Projekt mit einem Hauptprogramm erzeugt werden.

Alle Funktionen müssen in der Datei implementiert werden, in dem sich das Hauptprogramm befindet. Für jede Funktion muss ein Aufruf im Hauptprogramm gemacht werden.

2) Für jede Funktion muss vor der Implementierung der Funktion eine **Beschreibung** (Leistungsbeschreibung) mit dem im Unterricht verwendeten Schema gemacht werden.

AUFGABEN

1) Schreiben Sie die Funktion *volumen*, die das Volumen eines Quaders bestimmt.

2) Schreiben Sie die Funktion *mittelwert*, die **entweder** den arithmetischen Mittelwert oder den harmonischen Mittelwert zweier Zahlen bestimmt und außerdem zurückliefert, ob diese gleich groß sind. Ein Output davon soll über return realisiert werden.

3) Schreiben Sie die Funktion *feldmax*, die von zwei eindimensionalen Feldern (der Länge 2) in einem dritten eindimensionalen Feld (der Länge 2) in der ersten Zelle die Summe der beiden Zellen des ersten Feldes und in der zweiten Zelle des dritten Feldes die Differenz der beiden Zellen des zweiten Feldes abspeichert.

Beispiel:

1. Feld: [10, 20]

2. Feld: [30, 40]

3. Feld: [10 + 20, 30 - 40]

KLAUSUR 4 Programmierpraktikum 2BK11 23.06.2003 Zeit: 60 Minuten

Gruppe B Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

Bemerkung:

1) Es muss genau ein Projekt mit einem Hauptprogramm erzeugt werden.

Alle Funktionen müssen in der Datei implementiert werden, in dem sich das Hauptprogramm befindet. Für jede Funktion muss ein Aufruf im Hauptprogramm gemacht werden.

2) Für jede Funktion muss vor der Implementierung der Funktion eine **Beschreibung** (Leistungsbeschreibung) mit dem im Unterricht verwendeten Schema gemacht werden.

AUFGABEN

1) Schreiben Sie die Funktion *euro_dollar*, die aus dem Eurobetrag und dem Umrechnungsfaktor (mit dem man den Eurobetrag multiplizieren muss) den Dollarbetrag bestimmt.

2) Schreiben Sie die Funktion *brutto_netto*, die aus einem Betrag, der Mehrwertsteuer in Prozent (z.B. 16) und einem Modus **entweder** den Bruttopreis oder den Nettopreis bestimmt.

3) Schreiben Sie die Funktion *rechnen*, die von zwei eindimensionalen Feldern (der Länge 2) in einem dritten eindimensionalen Feld (der Länge 2) in der ersten Zelle die Summe aller Zellen des ersten und zweiten Feldes und in der zweiten Zelle des dritten Feldes das Produkt aller Zellen des ersten und zweiten Feldes bestimmt.

Beispiel:

1. Feld: [10, 20]

2. Feld: [30, 40]

3. Feld: [10 + 20 + 30 + 40 , 10 * 20 * 30 * 40]

KLAUSUR 4 Programmiertheorie 2BKI1 Nachtermin 1 Zeit: 60 Minuten

Name, Vorname:

Hilfsmittel:

selbstgeschriebene DIN A4-Seite

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Die Lösungen müssen als ablauffähige C-Programme dargestellt werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenoommen werden.
- Die rote Farbe darf nicht benutzt werden.
- Aufgabenblätter bitte auch abgeben.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

Bemerkung:

1) Es muss genau ein Projekt mit einem Hauptprogramm erzeugt werden.

Alle Funktionen müssen in der Datei implementiert werden, in dem sich das Hauptprogramm befindet. Für jede Funktion muss ein Aufruf im Hauptprogramm gemacht werden.

2) Für jede Funktion muss vor der Implementierung der Funktion eine **Beschreibung** (Leistungsbeschreibung) mit dem im Unterricht verwendeten Schema gemacht werden.

AUFGABEN

1) Schreiben Sie die Funktion *potenzen*, die das Quadrat und das Kubik zweier Zahlen bestimmt und außerdem zurückliefert, ob das Quadrat und das Kubik gleich groß sind.

2) Schreiben Sie eine Funktion, die einen Kleinbuchstaben in einen Grossbuchstaben und einen Grossbuchstaben in einen Kleinbuchstaben umwandelt. Außerdem soll die Funktion -1 zurückliefern, falls es sich nicht um einen Buchstaben handelt und sonst 0 zurückliefern.

3) Definition:

Die Fakultät einer natürlichen Zahl n ist wie folgt definiert:

$$n! = n * (n-1) * (n-2) * \dots * 1$$

Beispiel: $4! = 4 * 3 * 2 * 1$

Schreiben Sie die Funktion *fak*, die von einer natürlichen Zahl n deren Fakultät (also $n!$) bestimmt.

KLAUSUR 4 Programmierpraktikum 2BK11 Nachtermin 1 Zeit: 60 Minuten

Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

Bemerkung:

1) Es muss genau ein Projekt mit einem Hauptprogramm erzeugt werden.

Alle Funktionen müssen in der Datei implementiert werden, in dem sich das Hauptprogramm befindet. Für jede Funktion muss ein Aufruf im Hauptprogramm gemacht werden.

2) Für jede Funktion muss vor der Implementierung der Funktion eine **Beschreibung** (Leistungsbeschreibung) mit dem im Unterricht verwendeten Schema gemacht werden.

AUFGABEN

1) Schreiben Sie die Funktion *potenz*, die von einer reellen Zahl a die Zahl a^n bestimmt (n ist eine ganze Zahl).

2) Schreiben Sie die Funktion *kodieren*, die einen string kodiert, indem sie jeden Buchstaben in den im Alphabet folgenden umwandelt.

Beispiel:

aber ---kodieren---> bcfs

3) Schreiben Sie die Funktion *dekodieren*, die einen in der vorigen Aufgabe kodierten string wieder dekodiert.

KLAUSUR 1 Programmierpraktikum 2BK11 17.11.2003 Zeit: 60 Minuten

Gruppe A Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

1) Schreiben Sie ein C-Programm (Glücksspiel), das 3 Zufallszahlen zwischen 1 und 6 erzeugt (und damit das dreimalige Werfen eines Würfels simuliert).

Der Anwender soll diese 3 Zahlen erraten, indem er 3 Zahlen (zwischen 1 und 6) über Tastatur eingibt. Es wird vorausgesetzt, daß der Anwender auch tatsächlich Zahlen zwischen 1 und 6 eingibt.

a) Das Programm soll (im Ausgabeteil) die 3 Zufallszahlen auf dem Bildschirm ausgeben.

b) Das Programm soll ermitteln und im Ausgabeteil auf dem Bildschirm ausgeben, ob der Tipp des Anwenders richtig (in der richtigen Reihenfolge an) war.

c) Das Programm soll ermitteln und im Ausgabeteil auf dem Bildschirm ausgeben, wie "gut" der Tipp des Anwenders war.

Die Güte des Tipps ist die Summe aller Abstände zwischen dem jeweiligen Tipp und der Zufallszahl.

Beispiel

Zufallszahlen: 2 5 1

Tipp: 4 1 6

Dann ist die Güte: $(4-2) + (5-1) + (6-1) = 2 + 4 + 5 = 11$

d) Das Programm soll ermitteln und im Ausgabeteil auf dem Bildschirm ausgeben, wieviel Zahlen vom Anwender richtig getippt wurden.

KLAUSUR 1 Programmierpraktikum 2BK11 17.11.2003 Zeit: 60 Minuten
Gruppe B *Name, Vorname:*

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

1) Schreiben Sie ein C-Programm (Glücksspiel), das 3 Zufallszahlen zwischen 1 und 6 erzeugt (und damit das dreimalige Werfen eines Würfels simuliert).

Der Anwender soll diese 3 Zahlen erraten, indem er 3 Zahlen (zwischen 1 und 6) über Tastatur eingibt. Es wird vorausgesetzt, daß der Anwender auch tatsächlich Zahlen zwischen 1 und 6 eingibt.

a) Das Programm soll (im Ausgabeteil) die 3 Zufallszahlen auf dem Bildschirm ausgeben.

b) Das Programm soll ermitteln und im Ausgabeteil auf dem Bildschirm ausgeben, ob ein 3-erPasch (d.h. 3 gleiche Zahlen) gewürfelt wurde.

c) Wenn man die 3 Zufallszahlen bzw. die 3 Tipps nebeneinander schreibt, kann man diese als eine Zahl auffassen.

Das Programm soll ermitteln und im Ausgabeteil auf dem Bildschirm ausgeben, ob die als eine Zahl aufgefasste 3 Tipps größer ist als die als eine Zahl aufgefassten 3 Zufallszahlen.

Beispiel:

Zufallszahlen: 2 5 1 kann als Zahl 251 aufgefasst werden

Tipp: 4 1 6 kann als Zahl 416 aufgefasst werden

d) Das Programm soll ermitteln und im Ausgabeteil auf dem Bildschirm ausgeben, wieviel gerade Zahlen in dem Tipp vorkommen.

Aufgaben in Reserve

c) Das Programm soll ermitteln und im Ausgabeteil auf dem Bildschirm ausgeben, ob die Summe der Zufallszahlen gleich der Summe der getippten Zahlen ist.

Name, Vorname:

Hilfsmittel:

Ein DIN A4-Blatt selbst verfasste, handschriftliche Notizen

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punktabzug !!!!

AUFGABEN

1) Was gibt der folgende Programmausschnitt auf dem Bildschirm aus ? Begründen Sie!

```
int z;  
z=20;  
if(z=1)  
    printf("Die Zahl ist 1\n");  
else  
    printf("Die Zahl ist ungleich 1\n");
```

2) Der folgende, syntaktisch korrekte Programmausschnitt soll feststellen, ob 3 über Tastatur eingegebene ganzen Zahlen gleich sind, oder nicht.

a) Geben Sie 3 **konkrete**, ganzzahlige Werte für z1, z2 und z3 an, für die das Programm eine falsche Ausgabe erzeugt. Begründen Sie.

```
...  
if(z1==z2==z3)  
    printf("alle Zahlen sind gleich\n");  
else  
    printf("nicht alle Zahlen sind gleich\n");  
...
```

b) Ändern Sie das Programm so ab, dass es semantisch korrekt wird (das oben Angegebene macht).

3) Der folgende, syntaktisch korrekte Programmausschnitt soll feststellen, ob eine über Tastatur eingegebene ganze Zahl Element der folgenden Menge ist: {2; 4; 6; 8}:

a) Geben Sie einen **konkreten** Wert für z an, für den das Programm eine falsche Ausgabe erzeugt. Begründen Sie.

Bemerkung: Es wird vorausgesetzt, dass der Benutzer auch eine ganze Zahl eingibt.

Programmausschnitt:

```
...  
if(z==(2||4||6||8))  
    printf("Die Zahl ist entweder 2, 4, 6, oder 8");  
else  
    printf("Die Zahl ist nicht 2, 4, 6, oder 8");  
...
```

b) Ändern Sie das Programm so ab, dass es semantisch korrekt wird (das oben Angegebene macht).

4) Erstellen Sie von folgender IF...ELSE...IF-Kette (Programmteil) ein Struktogramm:

```
...  
if (bed < 0) {  
    x = 1;  
}  
else if (bed < 10) {  
    x = 2;  
}  
else if (bed < 20) {  
    x = 3;  
}  
else {  
    x = 4;  
}
```

5) Erstellen Sie ein Struktogramm zu dem Programm, das das Volumen und die Oberfläche eines Quaders mit den Seiten a, b und c berechnet.

Die Seitenlängen (Einheit: cm) müssen über Tastatur eingegeben werden.

Wenn für eine Seitenlänge ein Wert kleiner oder gleich 0 eingegeben wird, kann der Benutzer keine weitere Seitenlänge mehr eingeben und außerdem wird dann auch kein Volumen bzw. Oberfläche mehr berechnet, sondern eine entsprechende Meldung ausgegeben.

Nur wenn alle 3 Seitenlängen grösser Null sind, soll das Volumen und die Oberfläche berechnet werden.

Reserveaufgaben:

x)

--> if(a==b&&b==c)

...

&& bindet stärker als ==

x)

3) Der folgende, syntaktisch korrekte Programmausschnitt soll feststellen, ob eine über Tastatur eingegebene ganze Zahl Element der folgenden Menge ist: {2; 4; 6; 8}:

a) Geben Sie einen **konkreten** Wert für z an, für den das Programm eine falsche Ausgabe erzeugt. Begründen Sie.

Bemerkung: Es wird vorausgesetzt, dass der Benutzer auch eine ganze Zahl eingibt.

Programmausschnitt:

...

if(z==2||4||6||8)

printf("Die Zahl ist entweder 2, 4, 6, oder 8");

else

printf("Die Zahl ist nicht 2, 4, 6, oder 8");

...

Bemerkung:

Beachten Sie, daß der Operator == eine höhere Priorität hat als der Operator ||

KLAUSUR 2 Programmierpraktikum 2BKI1 15.12.2003 Zeit: 45 Minuten
Gruppe B *Name, Vorname:*

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

Schreiben Sie ein Programm, das die Potenz a^n einer reellen Zahl a bestimmt.
 n muß eine **ganze** Zahl sein.

Definition der Potenz:

$$a^n = \underbrace{a \cdot a \cdot \dots \cdot a}_{n \text{ - mal}}, \text{ wenn } n > 0$$

$$a^n = 1, \text{ wenn } n = 0$$

$$a^{-n} = \frac{1}{a^n}, \text{ wenn } n < 0$$

Beispiele:

$$2^4 = 2 \cdot 2 \cdot 2 \cdot 2 = 16$$

$$5^0 = 1$$

$$3^{-2} = \frac{1}{3^2} = \frac{1}{9}$$

KLAUSUR 2 Programmierpraktikum 2BK11 16.12.2003 Zeit: 45 Minuten
Gruppe A Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

Entwickeln Sie ein Programm, welches nach Eingeben des Startkapitals, des Sparziel und dem Sparzinssatz (den die Bank pro Jahr gibt) die Sparzeit (Anzahl der Jahre) berechnet und ausgibt, die nötig ist, um das Sparziel zu erreichen. Das tatsächlich erreichte Endkapital soll auch noch auf dem Bildschirm ausgegeben werden.

Die Zinsen bleiben nach jedem Jahr auf der Bank und werden nicht vom Anleger abgehoben.

Mathematische Anleitung anhand eines Beispiels (1 Zeitabschnitt = 1 Jahr):

Eingaben:

$K_0 = 100$ Euro; $p = 10\%$; Sparziel: 150 Euro

Berechnungen:

$p = 10\%$ Zinssatz ergibt ein Wachstumsfaktor von $q = 1 + \frac{10}{100} = 1,1$ jährlich.

Damit entwickelt sich das Kapital nach n Jahren wie folgt:

n	K_n (Kapital nach n Jahren in Euro)
0	100
1	$100 \cdot 1,1 = 111$
2	$100 \cdot 1,1 \cdot 1,1 = 121$
3	$100 \cdot 1,1 \cdot 1,1 \cdot 1,1 = 133,1$
4	$100 \cdot 1,1 \cdot 1,1 \cdot 1,1 \cdot 1,1 = 146,41$
5	$100 \cdot 1,1 \cdot 1,1 \cdot 1,1 \cdot 1,1 \cdot 1,1 = 161,051$

Ergebnis:

Sparzeit = 5 Jahre,

Endkapital = 161,051 Euro

Name, Vorname:

Hilfsmittel:

Eine DIN-A4 Seite selbstverfasster, handschriftlicher Notizen.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programnteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

AUFGABEN

1) Was ist allgemein der Hauptunterschied zwischen einer While-Anweisung und einer Do-Anweisung ?

Geben Sie dazu jeweils ein **konkretes** Beispiel (Programmteil) an.

2) Gegeben ist der folgende syntaktisch korrekte Programmteil:

```
i=0;
while (i<=3) ; {
    i = i+1;
    printf("Hallo Welt\n");
}
```

Wie oft gibt dieser Programmteil die Meldung "Hallo Welt" auf dem Bildschirm aus ?
Begründen Sie !

3) Gegeben ist der folgende Programmteil:

```
sum = 0;
i = 1;

do{
    sum = sum +i;
    i = i+1;
}
--> while (1==1)
```

a)

Tragen Sie die Werte der Variablen (an der Stelle -->) i und sum bei den ersten drei Schleifendurchgängen in die Tabelle ein (ohne Berücksichtigung der Schleifenbedingung)

i					
sum					

b)

Verändern Sie den Programmausschnitt an genau 2 Stellen so, daß der Wert der Variablen sum nach Verlassen der Schleife gleich der Summe $5 + 6 + 7 + \dots + 13 + 14 + 15$ ist.

4) Gegeben ist der folgende syntaktisch korrekte Programmteil (z1, z2 sind ganze Zahlen):

```
scanf ("%d", &z1);
scanf ("%d", &z2);
sum = 0;
i = z1;
while (i<=z2) {
    sum = sum+i;
    i = i+1;
}
```

a) Was berechnet der Programmteil (Bedingungen angeben, wann was geschieht) ?

b) Wandeln Sie aus dem Programmteil genau (nur) die While-Schleife in eine Do-While-Schleife um.

5)

Wandeln Sie die folgende For-Anweisung in eine Do-While-Anweisung um:

```
for (A1; B; A3) {
    A;
}
```

6)

Das folgende syntaktisch korrekte Programm soll die Summe aller ganzen Zahlen im Abstand zwei zwischen 2 vorgegebenen ganzen Zahlen berechnen.

```
void main() {
    int i, sum, z1, z2;
    sum = 0;
    printf("1. Zahl eingeben\n");
    scanf("%d", &z1);
    printf("2. Zahl eingeben\n");
    scanf("%d", &z2);

    i = z1;
    while(i!=z2+2) {
        sum = sum + i;
        i = i + 2;
    }

    printf("Summe = %d\n", sum);
}
```

Beispiele:

$z1 = 15, z2 = 23 \implies \text{Summe} = 15 + 17 + 19 + 21 + 23$

$z1 = 6, z2 = 8 \implies \text{Summe} = 6 + 8$

$z1 = 46, z2 = 46 \implies \text{Summe} = 46$

Geben Sie **konkret** einen Wert für z1 bzw. z2 an, (ganze Zahlen im Bereich des Datentyp int), für den das Programm zu einem "Absturz" führt. Begründen Sie !

Lösungen:

1) Die Do-Anweisung wird im Gegensatz zur While-Anweisung mindestens einmal durchlaufen.

```
i =10;
do{
    printf("%d ",i);
}
while(i<10);
```

```
i =10;
while(i<10){
    printf("%d ",i);
}
while(i<10);
```

2)

Null Mal, weil die While-Anweisung (Semikolon beachten !)

```
while (i<=3);
eine Endlos-Schleife ist
```

3)

a)

i	2	3	4		
sum	1	1 + 2	1 + 2 + 3		

b)

```
sum = 0;
i = 5;

do{
    sum = sum +i;
    i = i+1;
}
```

--> while (i<=15);

4)

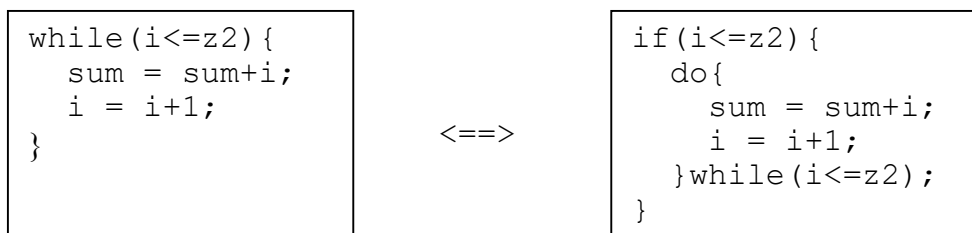
a)

Fall $z1 \leq z2$: Die Summe zwischen $z1$ und $z2$ (je einschließlich) wird berechnet

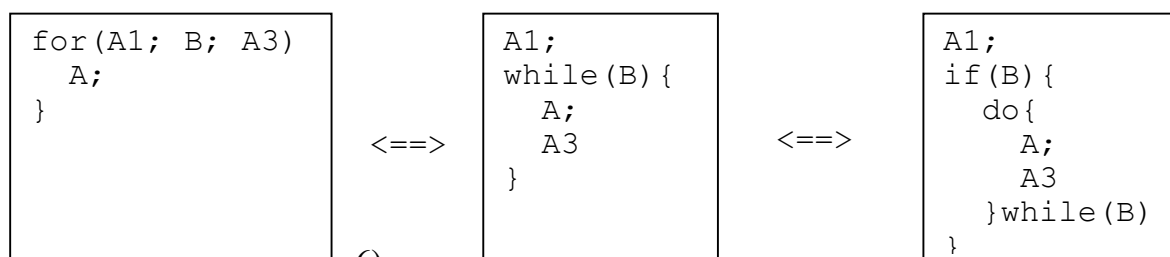
Beispiel: $z1 = 3, z2 = 7 \implies \text{Summe} = 3 + 4 + 5 + 6 + 7$

Fall $z1 > z2$: Die Summe ist 0.

b)



5)



6)

1. Beispiel: $z_1 = 9, z_2 = 3 \implies$ Endlos-Schleife, weil 9 bei jedem Durchgang um 2 erhöht wird und deshalb nie 3 wird.
2. Beispiel: $z_1 = 5, z_2 = 100 \implies$ Endlos-Schleife, weil 5 (ungerade Zahl) bei jedem Durchgang um 2 erhöht wird (und deshalb ungerade bleibt) und nie 100 werden kann, weil 100 eine gerade Zahl ist.

KLAUSUR 2 Programmiertheorie 2BKI1 Nachtermin 1 Zeit: 45 Minuten

Name, Vorname:

Hilfsmittel:

Keine

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punktabzug !!!!

AUFGABEN

1) Erstellen Sie ein Struktogramm (kein Programm !) zu dem Programm, das die Fakultät einer ganzzahligen Zahl (≥ 0) berechnet.

$$n! = n * (n-1) * (n-2) * \dots * 1$$

$$\text{Beispiele: } 1! = 1 \quad 2! = 2 * 1 \quad 3! = 3 * 2 * 1 \quad 4! = 4 * 3 * 2 * 1$$

Sie geben über Tastatur eine ganzzahlige Zahl ein. Von dieser Zahl soll die Fakultät berechnet werden und das Ergebnis auf dem Bildschirm ausgegeben werden.

Beispiel:

Sie geben z.B. die Zahl 4 ein. Auf dem Bildschirm wird dann ausgegeben:

$$n! = 24$$

Bemerkungen:

- a) $0!$ wird definiert als 1, also konkret: $0! = 1$
- b) Die Fakultät einer negativen Zahl ist nicht definiert.
- c) In dem Struktogramm muß berücksichtigt werden, daß der Anwender alle möglichen (auch negative) ganzzahligen Werte eingeben kann.

KLAUSUR 3 Programmierpraktikum 2BK11 8.3.2004 Zeit: 45 Minuten
Gruppe A Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

1) Entwickeln Sie ein Programm, welches Zahlen wie folgt kodiert:

- a) Lesen Sie Zufallszahlen (je einschließlich zwischen 1 und 6) in das Feld felda
- b) Geben Sie die Zahlen von felda auf dem Bildschirm (beginnend mit der nullten Zelle des Feldes) aus.
- c) Kopieren Sie die Zahlen von felda in das Feld felddb (beginnend mit der nullten Zelle des Feldes)
- d) Die Zahlen in felddb sollen in umgekehrter Reihenfolge in felda kopiert werden.

Beispiel:

felda (vorher) :

5	1	6	3	5
---	---	---	---	---

felda (nachher) :

5	3	6	1	5
---	---	---	---	---

- e) Geben Sie die Zahlen von felda auf dem Bildschirm (beginnend mit der nullten Zelle des Feldes) aus.

Bemerkungen:

- 1) Die Feldlänge soll vom Programmierer ohne aufwändige Bemühungen an genau einer Stelle leicht zu verändern sein.
(und muß dann neu kompiliert werden) und soll zuerst einmal den Wert 5 haben.
- 2) Jede Teilaufgabe entspricht einem Programmteil. Die ganze Aufgabe soll durch ein Projekt realisiert werden.

KLAUSUR 3 Programmierpraktikum 2BK11 8.3.2004 Zeit: 45 Minuten
Gruppe B *Name, Vorname:*

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

1) Entwickeln Sie ein Programm, welches Zahlen wie folgt kodiert:

- a) Lesen Sie Zufallszahlen (je einschließlich zwischen 1 und 6) in das Feld felda
- b) Geben Sie die Zahlen von felda auf dem Bildschirm (beginnend mit der nullten Zelle des Feldes) aus.
- c) Kopieren Sie die Zahlen von felda in das Feld felddb (beginnend mit der nullten Zelle des Feldes). Dabei soll jede Zahl zweimal hintereinander kopiert werden:
Beispiel:

felda :

5	1	6	3	5
---	---	---	---	---

felddb :

5	5	1	1	6	6	3	3	5	5
---	---	---	---	---	---	---	---	---	---

- d) Geben Sie die Zahlen von felddb auf dem Bildschirm (beginnend mit der nullten Zelle des Feldes) aus.

Bemerkungen:

- 1) Die Feldlänge soll vom Programmierer ohne aufwändige Bemühungen an genau einer Stelle leicht zu verändern sein.
(und muß dann neu kompiliert werden) und soll zuerst einmal den Wert 5 haben.
- 2) Jede Teilaufgabe entspricht einem Programmteil. Die ganze Aufgabe soll durch ein Projekt realisiert werden.

Name, Vorname:

Hilfsmittel:

Eine DIN-A4 Seite selbstverfasster, handschriftlicher Notizen.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

AUFGABEN

1)

In einem C-Programm wurde folgende Variable deklariert:

```
char v[4];
```

Geben Sie Ihren Kommentar zu folgenden Anweisungen ab:

```
v[0] = 'v';  
v[4] = 'r';  
v[-1] = 'r';
```

2) Wieviel Speicherplatz (in Bytes) benötigt der durch die folgende Deklaration reservierte Speicher ? Begründen Sie !

```
double werte[10][20];
```

3) Ein Programm fordert einen Anwender auf, in 100 verschiedenen Eingabefenstern Angaben (z.B. Name, Vorname, Wohnort, usw.) zu seiner Person (in Form von Zeichenketten) zu machen.

Der dafür reservierte Speicher soll vom Programmierer am Programmanfang mit '\0' initialisiert werden. Sollen für das Programm 100 eindimensionale oder ein zweidimensionales Feld verwendet werden ?

Beurteilen und begründen Sie dies aus der Sicht des Programmierers, der möglichst wenig Programmcode schreiben will.

4) Es wird das folgende zweidimensionale Feld deklariert:

```
char v[3][5];
```

Was ist v[1] ? Exakte Beschreibung !

5) Ein stabiles Programm soll entwickelt werden, in dem ein Anwender die Widerstandswerte der Widerstände einer elektrischen Schaltung eingeben kann. Diese Widerstandswerte müssen in einem eindimensionalen Feld gespeichert werden.

Erzeugen Sie dazu ein Struktogramm !

Bemerkung:

- 1) Überlegen Sie sich, wie Sie mit der Eingabe von negativen Widerstandswerten umgehen. (es gibt in der Elektrotechnik- zumindest hier für uns - nur positive (>0) Widerstandswerte).
- 2) Das Programm muß stabil sein, d.h. es darf nicht "abstürzen".

Lösungen:

1)

```
v[0] = 'v';    // richtig
v[4] = 'r';    // falsch: nicht reservierter Speicher
v[-1] = 'r';   // falsch: das erste Feld beginnt bei 0
```

2)

```
10 * 20 * sizeof(double) = 1600 Byte
```

3)

Bei 100 eindimensionalen Feldern braucht man 100 Schleifen, dagegen ist bei einem zweidimensionalen Feld nur eine Schleife nötig.

4)

v[1] ist ein eindimensionales Feld, das aus den folgenden Elementen besteht:
v[1][0], v[1][1], v[1][2], v[1][3], v[1][4]

5)

```
#include "stdafx.h"
#include "stdio.h"
```

```
void main() {
    const int len=5;
    double widerstaende[len];
    int anz = 0;
    double temp;
    // Schleifenabbruch: beenden=1, sonst beenden=0
    // weiter in der Schleife: beenden = 0
    int beenden = 0;

    printf("Bitte maximal %d Widerstaende eingeben\n", len-1);
    do{
        printf("Widerstaende eingeben\n");
        scanf("%lf", &temp);
        if(temp<=0){
            widerstaende[anz]=0;
            beenden = 1;
        }
        else{
            if(anz==len-2){
                widerstaende[anz]=temp;
                widerstaende[anz+1]=0;
                beenden=1;
            }
            else{
                widerstaende[anz]=temp;
                anz++;
                beenden=0;
            }
        }
    } while (beenden==0);
}
```

KLAUSUR 3 Programmiertheorie 2BKI1 Nachtermin 1 Zeit: 45 Minuten

Name, Vorname:

Name, Vorname:

Hilfsmittel:

Eine DIN-A4 Seite selbstverfasster, handschriftlicher Notizen.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punktabzug !!!!

AUFGABEN

1) Erstellen Sie ein Struktogramm zu der Funktion, die die Fläche unter einer Parabel mit der zugehörigen Funktionsgleichung $f(x) = x^2$ im Intervall $[a, b]$ mit n gleich langen Teilintervallen durch das unten beschriebene Annäherungsverfahren (Summe von Rechtecken) berechnet.

Konkret:

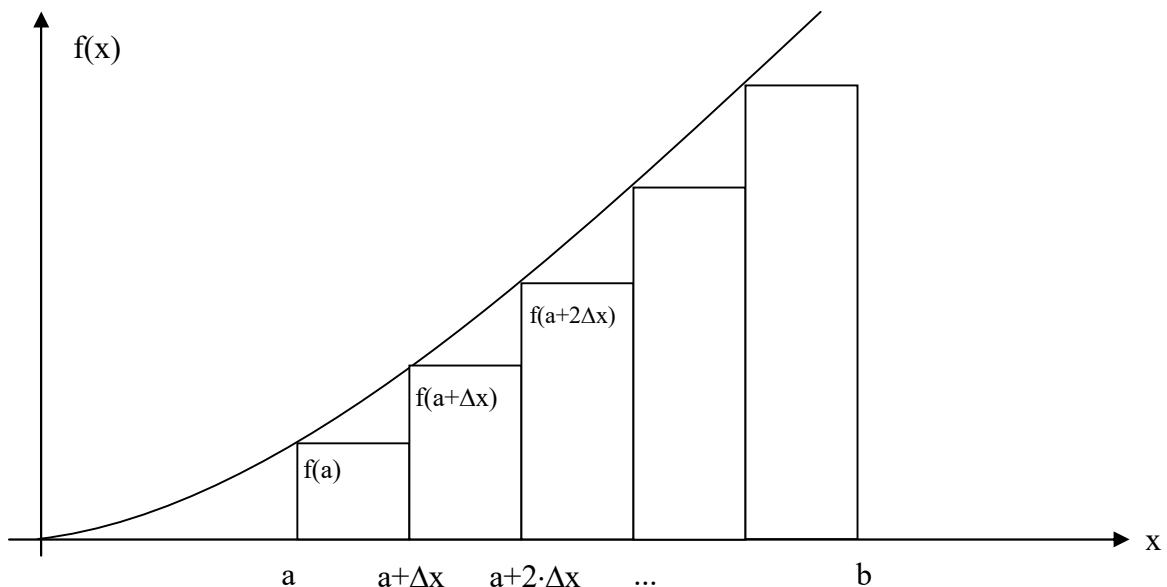
Man unterteilt das Intervall $[a, b]$ der Länge $b - a$ auf der x -Achse in n gleich lange Teilintervalle der Länge $\Delta x = (b-a) / n$.

Am Ende des jeweiligen Teilintervalls betrachtet man den Funktionswert an dieser Stelle und kann dann den Flächeninhalt des entsprechenden Rechtecks berechnen.

Beispiel:

Wenn man das Intervall $[a ; b]$ z.B. in $n = 5$ gleich lange Intervalle mit der Breite $\Delta x = (b-a)/n$ unterteilt, dann ist die Summe der Flächen der Rechtecke:

$$A = f(a) \cdot \Delta x + f(a+\Delta x) \cdot \Delta x + f(a+2\Delta x) \cdot \Delta x + f(a+3\Delta x) \cdot \Delta x + f(a+4\Delta x) \cdot \Delta x$$



KLAUSUR 3 Programmierpraktikum 2BKI1 Nachtermin 2 Zeit: 45 Minuten

Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

1) Schreiben Sie eine Funktion, die die Fläche unter einer Parabel mit der zugehörigen Funktionsgleichung $f(x) = x^2$ im Intervall $[a, b]$ mit n gleich langen Teilintervallen durch das unten beschriebene Annäherungsverfahren (Summe von Rechtecken) berechnet.

Konkret:

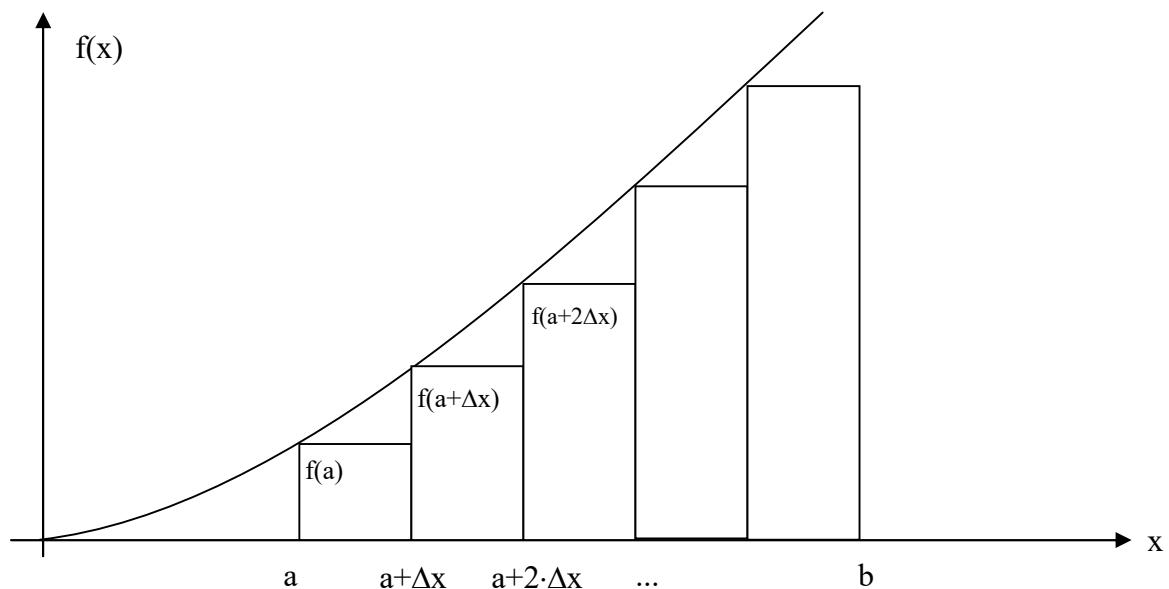
Man unterteilt das Intervall $[a, b]$ der Länge $b - a$ auf der x -Achse in n gleich lange Teilintervalle der Länge $\Delta x = (b-a) / n$.

Am Ende des jeweiligen Teilintervalls betrachtet man den Funktionswert an dieser Stelle und kann dann den Flächeninhalt des entsprechenden Rechtecks berechnen.

Beispiel:

Wenn man das Intervall $[a ; b]$ z.B. in $n = 5$ gleich lange Intervalle mit der Breite $\Delta x = (b-a)/n$ unterteilt, dann ist die Summe der Flächen der Rechtecke:

$$A = f(a) \cdot \Delta x + f(a+\Delta x) \cdot \Delta x + f(a+2\Delta x) \cdot \Delta x + f(a+3\Delta x) \cdot \Delta x + f(a+4\Delta x) \cdot \Delta x$$



KLAUSUR 4 Programmierpraktikum 2BK11 7.6.2004 Zeit: 45 Minuten
Gruppe A *Name, Vorname:*

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

1) Schreiben Sie die Funktion "kuerzen", die einen Bruch maximal kürzt:

Beispiel: $18 / 24 = 3 / 4$

Erzeugen Sie im Hauptprogramm einen Aufruf dieser Funktion.

Bemerkungen:

Vor der Implementierung der Funktion muß eine **Beschreibung** (Leistungsbeschreibung) mit dem im Unterricht verwendeten Schema gemacht werden.

KLAUSUR 4 Programmierpraktikum 2BK11 8.6.2004 Zeit: 45 Minuten
Gruppe B Name, Vorname:

Hilfsmittel:

Bücher und Skripte eigener Wahl.

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!
- Bei praktischen Arbeiten am Computer bitte folgendes beachten:
- NUR **ablauffähige** C-Programme werden bewertet ! **Jede** Aufgabe als eigenes Projekt
- Der Name, Vorname und die Aufgabennummer (z.B: Aufgabe 1) muß als **Kommentar** in jedes Programm mitaufgenommen werden.
- Name der Quelldatei nach folgendem Schema vergeben: Gruppe_Rechnernummer_Nachname_Vorname_Aufgabennr.
Beispiel: A_12_Mustermann_Erika_nr3
- Die Programme müssen ausgedruckt werden.
- Die entsprechenden Quelldateien müssen auf Diskette abgespeichert und abgegeben werden.

AUFGABEN

1) Schreiben Sie die Funktion *kodieren*, die einen String kodiert, indem sie jeden Buchstaben in den im Alphabet folgenden umwandelt.

Erzeugen Sie im Hauptprogramm einen Aufruf dieser Funktion.

Beispiel:

String vor der Kodierung:

v	i	e	l	e
---	---	---	---	---

String nach der Kodierung:

w	j	f	m	f
---	---	---	---	---

Bemerkungen:

Vor der Implementierung der Funktion muß eine **Beschreibung** (Leistungsbeschreibung) mit dem im Unterricht verwendeten Schema gemacht werden.

KLAUSUR 4 Programmiertheorie 2BKI1 17.6.2004 Zeit: 45 Minuten

Name, Vorname:

Hilfsmittel:

Keine

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punktabzug !!!!

AUFGABEN

1)

Geben Sie die syntaktischen Fehler in den folgenden Funktionen an.

Welche Werte haben die Variablen ii, ff, dd, usw. nach dem jeweiligen Aufruf im

Hauptprogramm main ? Geben Sie diese Werte jeweils im mit // bezeichnet Kommentar an.

```
void g1(int i){
    return(i);
}

void g2(float *f){
    f = *f;
}

void g3(double *d){
    *d = *d * *d;
}
```

```
void h1(int i){
    i = 2 * i;
}

void h2(float x, float *f){
    x = *f + 1;
}

double h3(double *d){
    double y;
    y = *d * *d;
    return(y);
}
```

```
void main(){
    int ii = 5;
    float ff = 4;
    float erg = 3;
    double dd = 2;

    ii = g1(2);           // ii =
    g2(&ff);              // ff =
    g3(&dd);              // dd =
    h1(ii);               // ii =
    h2(erg, &ff);         // erg =
    dd = h3(&dd);         // dd =
}
```

2)

a) Sie wollen ein Programm schreiben, das den Mittelwert einer Klassenarbeit (irgend einer beliebigen Klasse) berechnet.

Sie wollen dazu eine Funktion benutzen, die Sie aber wegen Arbeitsüberlastung von einem "Programmierknecht" implementieren (programmieren) lassen.

Entwerfen Sie für den "Programmierknecht" eine **Beschreibung** (Leistungsbeschreibung) dieser Funktion mit dem im Unterricht verwendeten Schema. (Kein Programm !!!)

b) Da der "Programmierknecht" gerade einen längeren Urlaub in der Südsee verbringt, sind Sie gezwungen nach der von Ihnen entworfenen Leistungsbeschreibung die Funktion selbst zu programmieren.

3)

Programmieren Sie anhand der folgenden Beschreibung die dazugehörige Funktion:

```
/*
**
**  int benzin(double km, double l, double lp,
**             double *l100, double *e100)
**
**
**#*****
/*
```

Parameter:

(i) double km: Strecke (in Km)
(i) double l: Verbrauch (in Liter)
(i) double lp: Literpreis Benzin
(o) double l100: Benzinverbrauch in Liter pro 100 Km
(o) double e100: Benzinpreis in Euro pro 100 Km

Return:

-1: km <= 0
 0: sonst

Beschreibung:

Berechnet in Abhängigkeit von den gefahrenen Kilometern (km), den dafür benötigten Litern Benzin (l) und dem Benzinliterpreis (lp) den Literverbrauch Benzin pro 100 Km (l100) und die Kosten in Euro für das Benzin pro 100 Km (e100).

Falls eine Kilometerzahl <= 0 angegeben wird, wird -1 zurückgegeben, ansonsten 0.

*/

KLAUSUR 4 Programmiertheorie 2BKI1 Nachtermin 1 Zeit: 45 Minuten

Name, Vorname:

Hilfsmittel:

Keine

Hinweise (unbedingt beachten):

- Alle Aufgaben müssen bearbeitet werden.
- Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- Aufgabenblätter bitte auch abgeben.
- Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- Die rote Farbe darf nicht benutzt werden.
- Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- Selbsterklärende Variablennamen benutzen.
- Programme müssen benutzerfreundlich sein.
- EVA-Prinzip muss benutzt werden.
- Einrücken der entsprechenden Programmteile.
- Bei Nichtbeachtung dieser Hinweise gibt es einen Punktabzug !!!!

AUFGABEN

1) a) Sie wollen ein Programm schreiben, das die beste und die schlechteste Note einer Klassenarbeit (irgend einer beliebigen Klasse) berechnet.
Sie wollen dazu eine Funktion benutzen, die Sie aber wegen Arbeitsüberlastung von einem "Programmierknecht" implementieren (programmieren) lassen.
Entwerfen Sie für den "Programmierknecht" eine **Beschreibung** (Leistungsbeschreibung) dieser Funktion mit dem im Unterricht verwendeten Schema. (Kein Programm !!!)

b) Da der "Programmierknecht" gerade einen ausgedehnten Urlaub in einem westeuropäischen Küstenstaat verbringt, sind Sie gezwungen nach der von Ihnen entworfenen Leistungsbeschreibung die Funktion selbst zu programmieren.

2) Programmieren Sie anhand der folgenden Beschreibung die dazugehörige Funktion:

```
/*  
**  
** void strcpy (char *str1, char *str2, int len) **  
**  
*#*****  
*/
```

Parameter:

- (i) char *str1: der kopierte String
- (i) char *str2: der zu kopierende String
- (i) int len: Länge des Strings

Return:

kein

Beschreibung:

Kopiert len Zeichen des Strings str2 (vom Anfang diesen Strings an) in den String str1 (vom Anfang diesen Strings an) und beendet den String str1 mit '\0'

Beispiel:

```
strcpy(str1, "Edelbert", 3);  
In str1 steht dann nach dem Aufruf: "Ede"
```

*/