Script **generated by TTT**

Title: Einf_HF (17.06.2013)

Date: Mon Jun 17 14:15:07 CEST 2013

Duration: 91:58 min

Pages: 38

"Kunst des Programmierens". Grundlagen zu Datenstrukturen, Programmkonstrukte, Strukturierung von Programmen, objekt-orientierte Programmierung.

- Fragestellungen des Abschnitts:
  - Was ist ein Algorithmus?
  - Welche elementaren Datenstrukturen gibt es?
  - Was sind die grundlegenden Konstrukte einer Programmiersprache?
  - Was ist unter Objekt-orientierter Programmierung zu verstehen?
  - Was versteht man unter Modularisierung und Rekursion?

Objekt: Exemplar (Instanz) eines Gegenstandes oder Begriffs. Möglichst stark an reale Welt angelehnt.

Objekt

Ein Objekt hat

1. einen ***eindeutigen Objektnamen*** , z.B. vom Benutzer vergebene Namen. Zusätzlich interner Identifikator.
2. einen ***Zustand*** , definiert durch Attribute und die zugehörigen Attributwerte ("Instanzvariable", private Daten).
3. ein ***Verhalten*** , spezifiziert durch eine Menge von Operationen (Methoden), die auf den Attributen agieren und Operationen anderer Objekte aufrufen können;
4. Beziehungen zu anderen Objekten.

Notation für den Zugriff auf die Objektdaten

Objektname.Attributname = Attributwert

Klasse

Objekte mit gleichen Attributen und gleichem Verhalten. Objekt "weiß", zu welcher Klasse es gehört.

Beispiel: ***Klasse*** Person, zu der die ***Objekte*** Schmidt, Mayer, Müller gehören.

Beispiel: Klassendefinition

```
public class circle {
    double x, y; // Koordinaten der Kreismitte
    double r; //Radius des Kreises
    // Konstruktor für die Erzeugung von Objekten:
    public circle (double xcoord, double ycoord, double radius)
       {x = xcoord; y = ycoord; r = radius;}
    // Methoden, die Umfang und die Fläche
    // als Ergebnis liefern:
    public double circumference(){return 2 * 3.14159 * r;}
    public double area(){return 3.14159 * r * r;}
}
```

`double` bezeichnet eine Variable mit doppelter Genauigkeit.

Schutzmechanismen

Kontrolle des Zugriffs auf Daten und Methoden von außen

public : Zugriff von außen möglich.

private : Zugriff nur von innerhalb der Klasse möglich.

Erzeugen eines Objekts

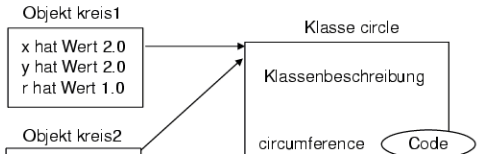
Nach Spezifikation einer Klasse: beliebig viele Objekte dazu erzeugbar ("Objektinstanziierung"). Jeweils eigene Attributwerte, jedoch Methoden gemeinsam.

Beispiel

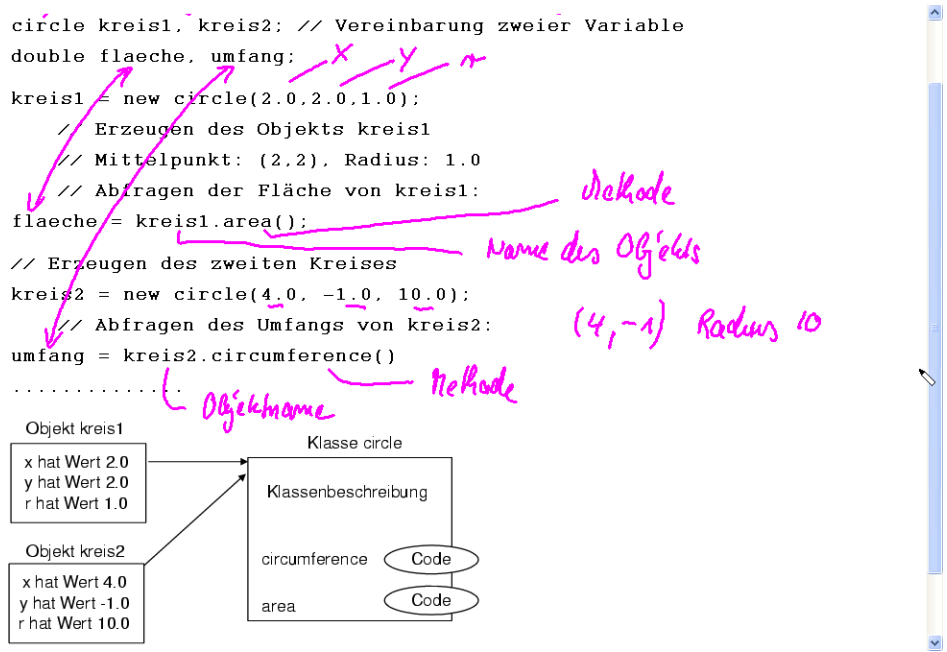
```
circle kreis1, kreis2; // Vereinbarung zweier Variable
double flaeche, umfang;

kreis1 = new circle(2.0,2.0,1.0);
    // Erzeugen des Objekts kreis1
    // Mittelpunkt: (2,2), Radius: 1.0
    // Abfragen der Fläche von kreis1:
flaeche = kreis1.area();

// Erzeugen des zweiten Kreises
kreis2 = new circle(4.0, -1.0, 10.0);
    // Abfragen des Umfangs von kreis2:
umfang = kreis2.circumference()
.............
```



Erzeugen eines Objekts



Vererbung

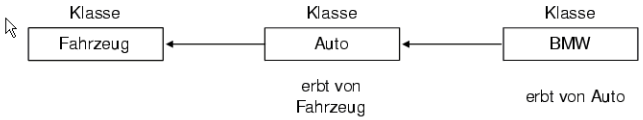
Werkzeug zum Organisieren und Konstruieren von Klassen; Wiederverwendung.

Definition - Vererbung

Seien K und K_i , i = 1,..,n Klassen; Vererbung ist eine Beziehung zwischen K und den K_i , wobei Struktur und Verhalten von K durch Struktur und Verhalten der K_i bestimmt wird; K "erbt" von den Klassen K_i ;

Beziehung zwischen Klassen; K Unterklasse (Subklasse) von K_i ; K_i Oberklasse (Superklasse) von K;

Unterklassen übernehmen Eigenschaften (Attribute, Operationen und Beziehungen) der Oberklasse(n); ggf. über mehrere Stufen. Betrifft Attribute und Methoden;



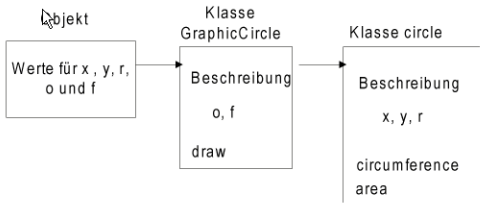
Einfachvererbung: eine Klasse hat nur eine direkte Oberklasse, d.h. n=1.

Mehrfachvererbung: eine Klasse hat mehrere direkte Oberklassen

Beispiel

Beispiel

```
public class GraphicCircle extends circle {
    // Es werden automatisch die Variablen und Methoden
    // der Klasse circle geerbt;
    // nur die neuen Informationen müssen hier
    // aufgeführt werden:
    // Die Farben für Rand und Füllfläche.
    color o, f;
    public GraphicCircle (double xcoord, double ycoord, double radius, color
    outline, color fill)
        {super(xcoord,ycoord,radius); o = outline; f = fill;}
    public void draw(Window dw)
        {dw.drawCircle(x, y, r, o, f;}
}
```



Objektorientierte Programmierung

Softwaresystem realisiert durch Menge von Objekten. Gegensatz prozedurale Programmierung: Anweisungen im Vordergrund.

Objektorientiertes Programmieren: Daten im Vordergrund. In Objekten zusammengefasst ("Verkapselung"). Funktionen lokal bei Objekten definiert.

Die Funktionen werden Methoden genannt.

Objekt - Klasse

Erzeugen eines Objekts

Vererbung

Grundlagen der Programmierung

"Kunst des Programmierens". Grundlagen zu Datenstrukturen, Programmkonstrukte, Strukturierung von Programmen, objekt-orientierte Programmierung.

- Fragestellungen des Abschnitts:
  - Was ist ein Algorithmus?
  - Welche elementaren Datenstrukturen gibt es?
  - Was sind die grundlegenden Konstrukte einer Programmiersprache?
  - Was ist unter Objekt-orientierter Programmierung zu verstehen?
  - Was versteht man unter Modularisierung und Rekursion?

Einführung

Algorithmus

Datentypen und Ausdrücke

Programmkonstrukte

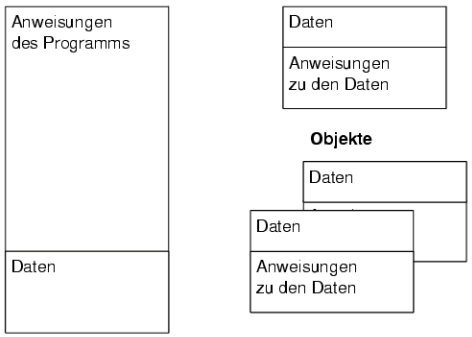
Objektorientierte Programmierung

Modularisierung von Programmen

Rekursion

Prozedurales / Objektorientieres Programmieren

Prozedurales Programmieren **Objektorientiertes Programmieren**



Prozedurales Programmieren

Sequenz von Anweisungen, ausführen, warten. Basiselemente: Sequenz, Auswahl (if-then-else), Iteration. Trennung von Daten und Anweisungen.

Objektorientiertes Programmieren

Objekte statt Anweisungen. Objekt hat Attribute, Methoden und Ereignisbehandlung. Zusammenfassung von Daten und Anweisungen

Modularisierung von Programmen

Systematischer Aufbau von Programmen durch Zerlegung in eigenständige Einheiten.

Allgemeines

Beispiel Zerlegung

Strukturierung von Algorithmen

Module

Prozedurales / Objektorientieres Programmieren

Entwurf von Algorithmen

Generell sinnvolle Vorgehensweise:

- Spezifikation der Problemstellung
- Bestimmung der Definitionsbereiche und der Datentypen
- Suche nach / Vergleich mit bekannten Algorithmen, eventuell Erweiterung oder Anpassung der bekannten Algorithmen
- Zerlegung des Problems in Teilprobleme ("divide and conquer"-Vorgehensweise); auf jeder Zerlegungsebene Verwendung von
  - Sequenz von Schritten
  - Fallunterscheidung (Alternativen)
  - Iteration von Schritten (Schleifen)
- Wiederhole diese Vorgehensweise für jedes Teilproblem

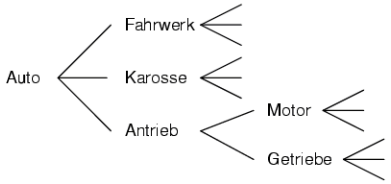
Schrittweises Verfeinern von Algorithmen

Direktes Vorgehen oft schwierig, da Problemumfang sehr groß. Oft nicht klar, welche zusätzlichen Aspekte mitzuberücksichtigen sind.

Problem: Unüberschaubarkeit bei komplexen Aufgaben

Lösung: Zuerst in größeren, gröberen Einheiten denken, dann diese zerlegen

Beispiel: Entwicklung eines Autos



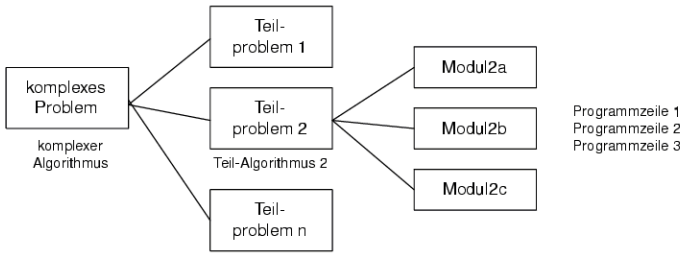
Übertragung dieses Ansatzes auf die Zerlegung von Algorithmen in kleinere und überschaubarere Einheiten.

Allgemeines

Entwurf von Algorithmen

Schrittweises Verfeinern von Algorithmen

Algorithmen-Zerlegung



Beispiel Zerlegung

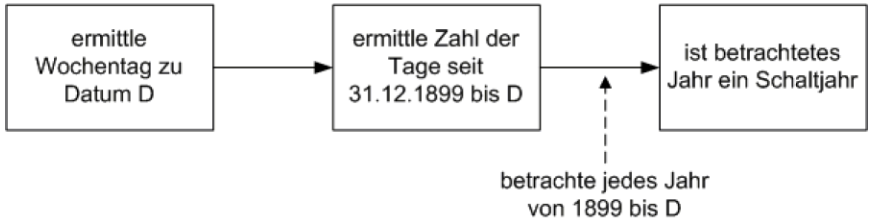
Aufgabenstellung: Ermittle zu einem gegebenen Datum nach dem 1.1.1900 (gegeben als Tag im Monat, Monat im Jahr und Jahr AD) den Wochentag.

Aufteilung der Aufgabe in mehrere Teilaufgaben

Teilaufgabe 1: Zahl der Tage seit dem 31.12.1899

Teilaufgabe 2: Ermittlung ob Schaltjahr (Modul)

Aufrufabhängigkeiten der Teilaufgaben



Beispiel für den 24.12.2000

Aufteilung der Aufgabe in mehrere Teilaufgaben

Der 31.12.1899 war ein Sonntag

Ermittle die Zahl der Tage zwischen dem 31.12.1899 und dem gegebenen Datum (Modul)

Dividiere diese Zahl durch 7 und ermittle den Rest der Division

Wenn der Rest 0 ist, dann ist das Datum ein Sonntag, bei Rest 1 ein Montag, bei Rest 2 ein Dienstag, ...

Teilaufgabe 1: Zahl der Tage seit dem 31.12.1899

Teilaufgabe wird als Modul mit folgendem Ablauf realisiert:

Setze Zähler auf 0

Addiere für jedes Jahr (vor dem aktuellen Jahr) seit 1900 die Zahl 365 zum Zähler, wenn Jahr ein Schaltjahr ist, dann addiere zusätzlich 1

Addiere für jeden vollendeten Monat im aktuellen Jahr die Zahl der Tage im Monat zum Zähler, wenn Monat Februar und Jahr Schaltjahr, dann addiere zusätzlich 1

Addiere die Zahl der vollendeten oder angebrochenen Tage im aktuellen Monat zum Zähler

Beispiel Zerlegung

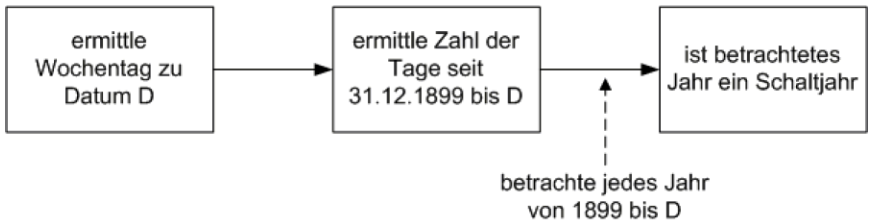
Aufgabenstellung: Ermittle zu einem gegebenen Datum nach dem 1.1.1900 (gegeben als Tag im Monat, Monat im Jahr und Jahr AD) den Wochentag.

Aufteilung der Aufgabe in mehrere Teilaufgaben

Teilaufgabe 1: Zahl der Tage seit dem 31.12.1899

Teilaufgabe 2: Ermittlung ob Schaltjahr (Modul)

Aufrufabhängigkeiten der Teilaufgaben



Beispiel für den 24.12.2000

Beispiel für den 24.12.2000

Zähler = 0

Gehe von 1900 bis 1999 und addiere jeweils 365 oder 366; Ergebnis ist 36524.

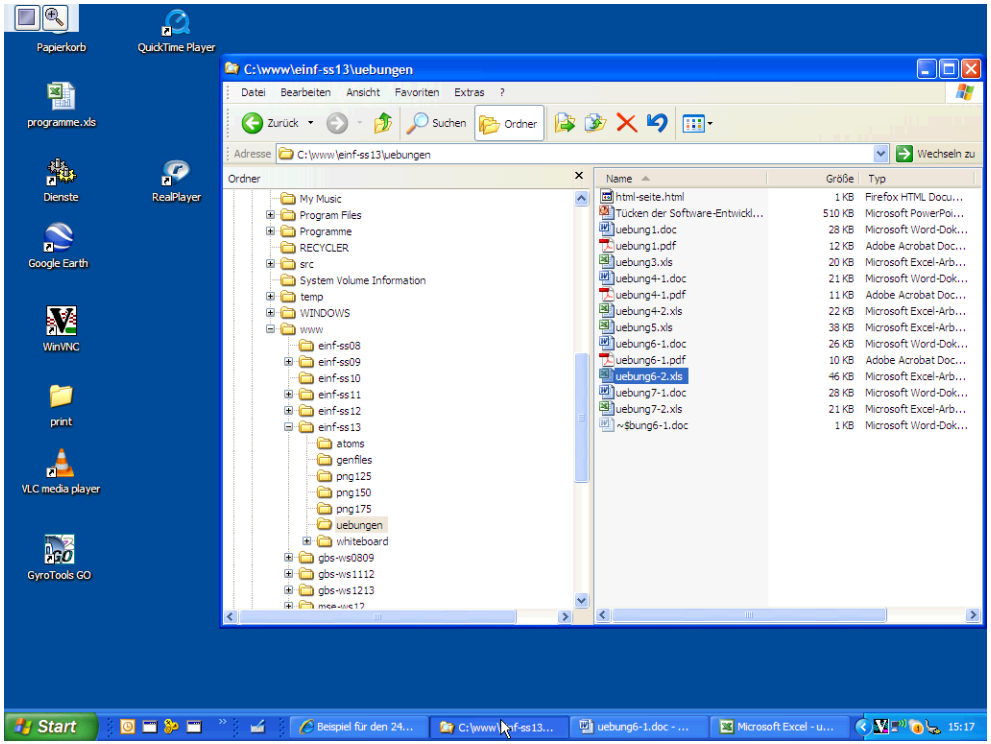
Gehe von Januar bis November und addiere 31, 28+1, 31, 30, 31, 30, 31, 31, 30, 31, 30 = 335

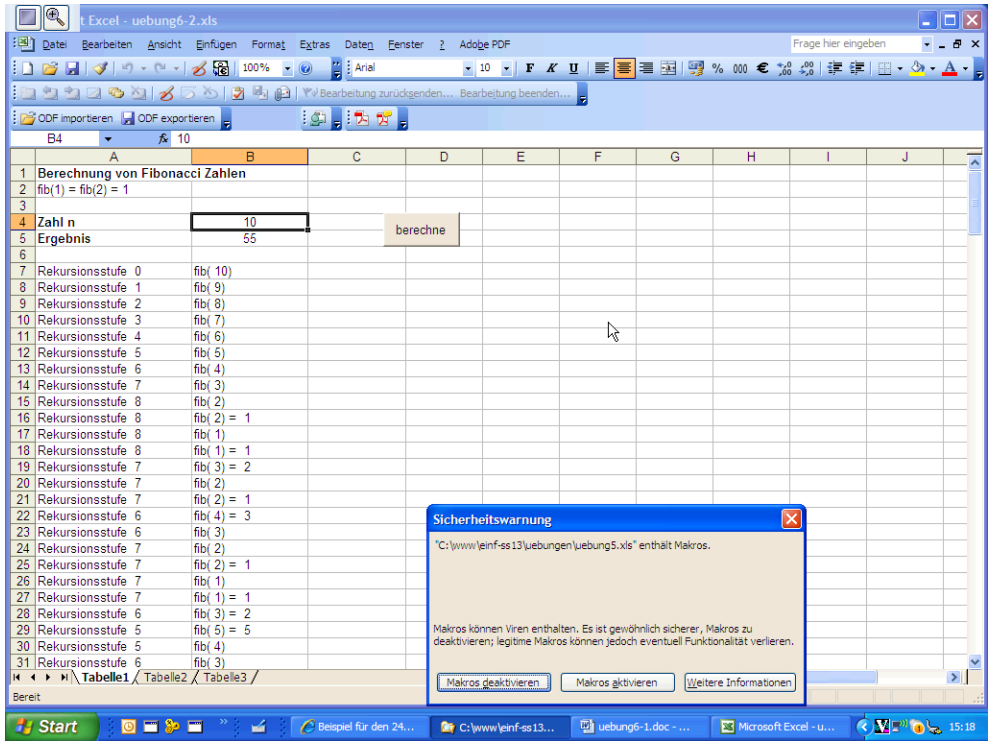
Addiere 24

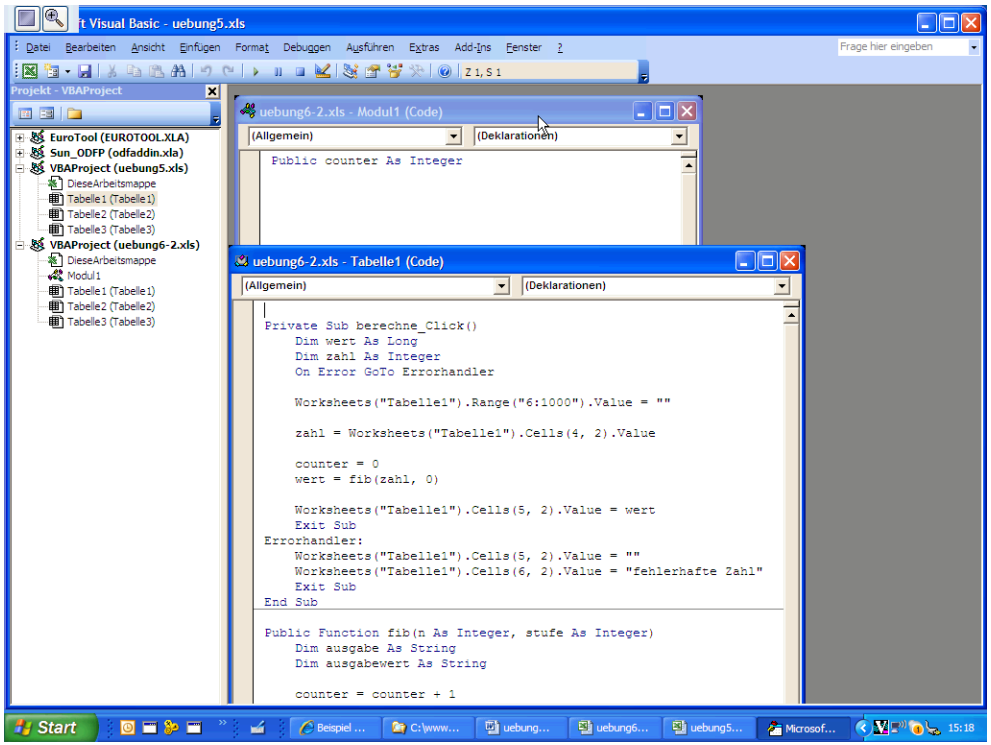
Zähler = 36883

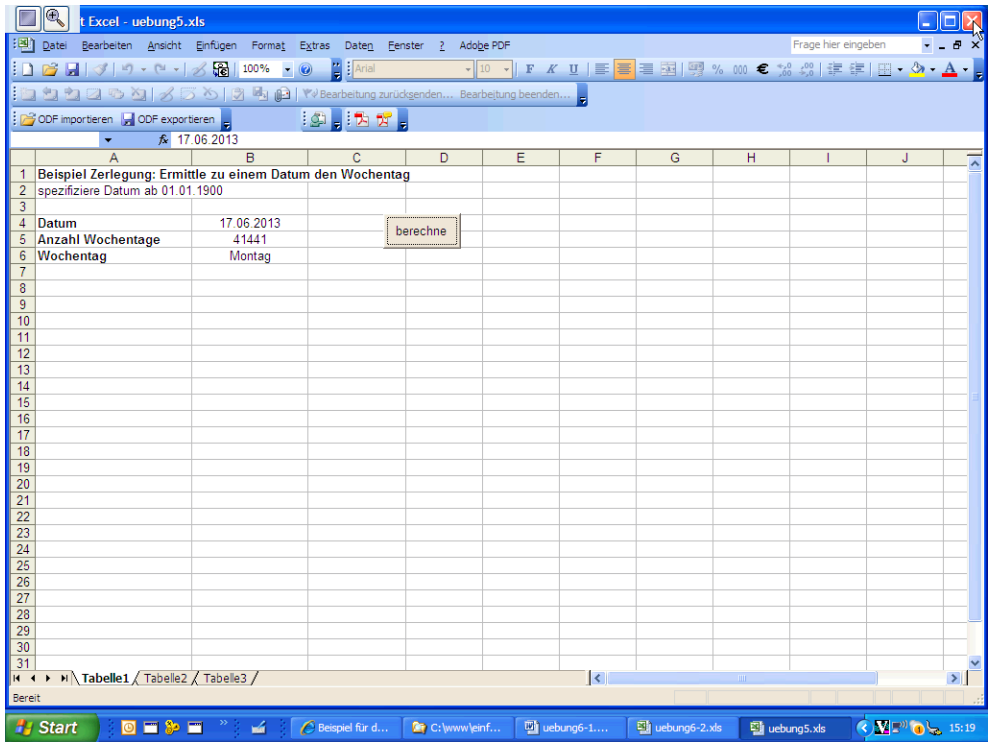
36883/7 = 5269

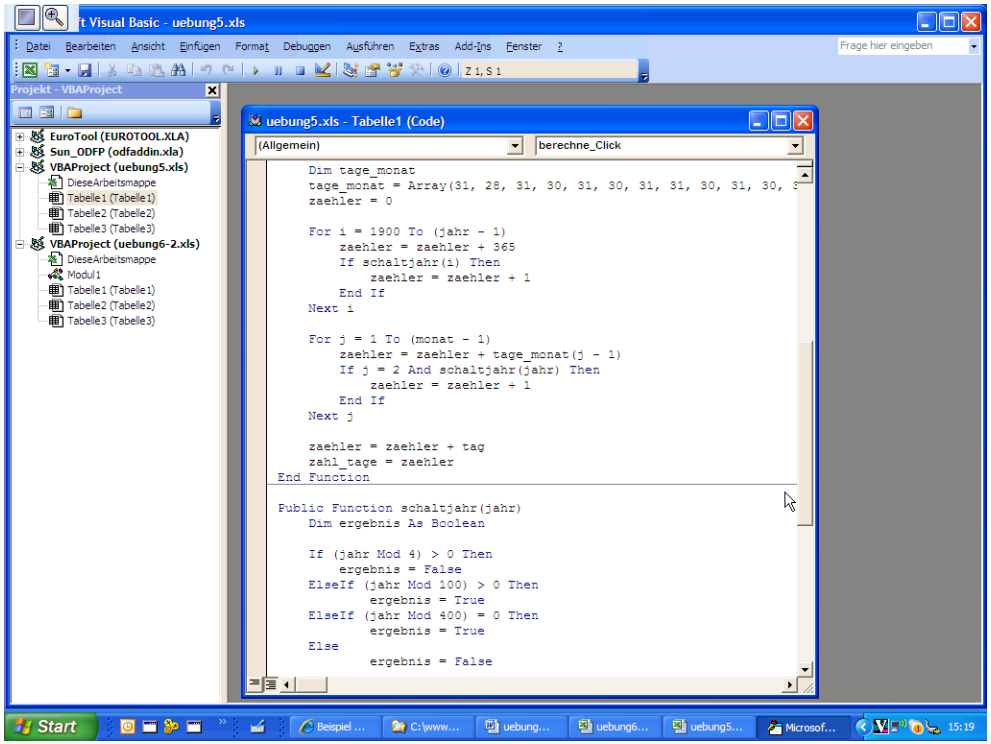
Rest der Division ist 0, also ist der 24.12.2000 ein Sonntag

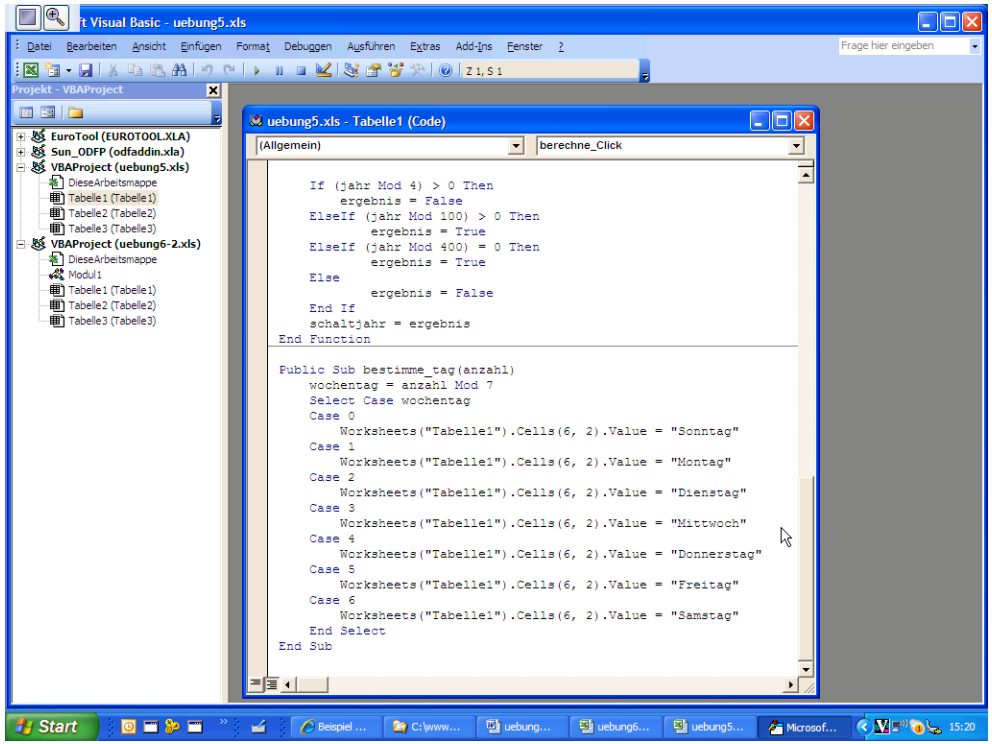

C:\www\einf-ss13\uebungen
Datei Bearbeiten Ansicht Favoriten Extras ?
Zurück Suchen Ordner
Adresse C:\www\einf-ss13\uebungen
Wechseln zu
Ordner
My Music
Program Files
Programme
RECYCLER
src
System Volume Information
temp
WINDOWS
www
einf-ss08
einf-ss09
einf-ss10
einf-ss11
einf-ss12
einf-ss13
atoms
genfiles
png125
png150
png175
uebungen
whiteboard
gbs-ws0809
gbs-ws1112
gbs-ws1213
Name
Größe
Typ
html-seite.html 1 KB Firefox HTML Docu...
Tücken der Software-Entwickl... 510 KB Microsoft PowerPoi...
uebung1.doc 28 KB Microsoft Word-Dok...
uebung1.pdf 12 KB Adobe Acrobat Doc...
uebung3.xls 20 KB Microsoft Excel-Arb...
uebung4-1.doc 21 KB Microsoft Word-Dok...
uebung4-1.pdf 11 KB Adobe Acrobat Doc...
uebung4-2.xls 22 KB Microsoft Excel-Arb...
uebung5.xls 38 KB Microsoft Excel-Arb...
uebung6-1.doc 26 KB Microsoft Word-Dok...
uebung6-1.pdf 10 KB Adobe Acrobat Doc...
uebung6-2.xls 46 KB Microsoft Excel-Arb...
uebung7-1.doc 28 KB Microsoft Word-Dok...
uebung7-2.xls 21 KB Microsoft Excel-Arb...
~$bung6-1.doc 1 KB Microsoft Word-Dok...
Papierkorb
QuickTime Player
programme.xls
Dienste
RealPlayer
Google Earth
WinVNC
print
VLC media player
GyroTools GO
Start
Beispiel für den 24...
C:\www\einf-ss13...
uebung6-1.doc - ...
Microsoft Excel - u...
15:17

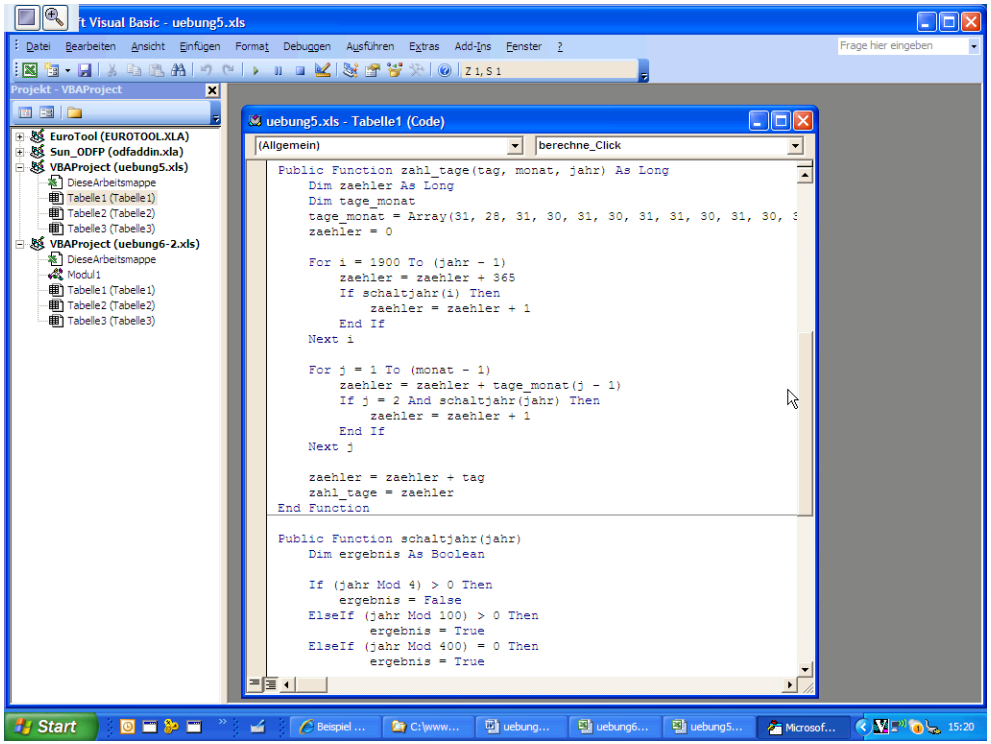

t Excel - uebung6-2.xls
Datei Bearbeiten Ansicht Einfügen Format Extras Daten Fenster ? Adobe PDF
Frage hier eingeben
ODF importieren ODF exportieren
B4 10
Berechnung von Fibonacci Zahlen
fib(1) = fib(2) = 1
Zahl n 10
Ergebnis 55
berechne
Rekursionsstufe 0 fib(10)
Rekursionsstufe 1 fib(9)
Rekursionsstufe 2 fib(8)
Rekursionsstufe 3 fib(7)
Rekursionsstufe 4 fib(6)
Rekursionsstufe 5 fib(5)
Rekursionsstufe 6 fib(4)
Rekursionsstufe 7 fib(3)
Rekursionsstufe 8 fib(2) = 1
Rekursionsstufe 8 fib(1)
Rekursionsstufe 8 fib(1) = 1
Rekursionsstufe 7 fib(3) = 2
Rekursionsstufe 7 fib(2)
Rekursionsstufe 7 fib(2) = 1
Rekursionsstufe 6 fib(4) = 3
Rekursionsstufe 6 fib(3)
Rekursionsstufe 7 fib(2)
Rekursionsstufe 7 fib(2) = 1
Rekursionsstufe 7 fib(1)
Rekursionsstufe 7 fib(1) = 1
Rekursionsstufe 6 fib(3) = 2
Rekursionsstufe 5 fib(5) = 5
Rekursionsstufe 5 fib(4)
Rekursionsstufe 6 fib(3)
Tabelle1 Tabelle2 Tabelle3
Bereit
Sicherheitswarnung
"C:\www\einf-ss13\uebungen\uebung5.xls" enthält Makros.
Makros können Viren enthalten. Es ist gewöhnlich sicherer, Makros zu deaktivieren; legitime Makros können jedoch eventuell Funktionalität verlieren.
Makros deaktivieren
Makros aktivieren
Weitere Informationen
Start
Beispiel für den 24...
C:\www\einf-ss13...
uebung6-1.doc - ...
Microsoft Excel - u...
15:18

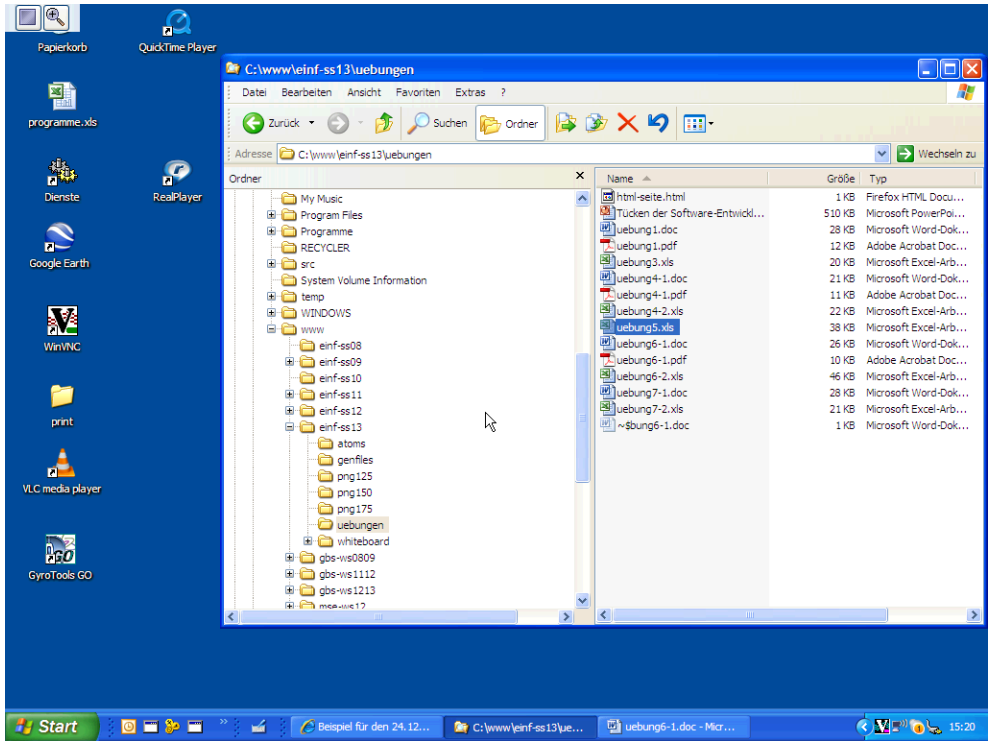

t Visual Basic - uebung5.xls
Datei Bearbeiten Ansicht Einfügen Format Debuggen Ausführen Extras Add-Ins Fenster ?
Frage hier eingeben
Projekt - VBAProject
EuroTool (EUROTOOL.XLA)
Sun_ODFP (odfaddin.xla)
VBAProject (uebung5.xls)
DieseArbeitsmappe
Tabelle1 (Tabelle1)
Tabelle2 (Tabelle2)
Tabelle3 (Tabelle3)
VBAProject (uebung6-2.xls)
DieseArbeitsmappe
Modul1
Tabelle1 (Tabelle1)
Tabelle2 (Tabelle2)
Tabelle3 (Tabelle3)
uebung6-2.xls - Modul1 (Code)
(Allgemein)
(Deklarationen)
Public counter As Integer
uebung6-2.xls - Tabelle1 (Code)
(Allgemein)
(Deklarationen)
Private Sub berechne_Click()
Dim wert As Long
Dim zahl As Integer
On Error GoTo Errorhandler
Worksheets("Tabelle1").Range("6:1000").Value = ""
zahl = Worksheets("Tabelle1").Cells(4, 2).Value
counter = 0
wert = fib(zahl, 0)
Worksheets("Tabelle1").Cells(5, 2).Value = wert
Exit Sub
Errorhandler:
Worksheets("Tabelle1").Cells(5, 2).Value = ""
Worksheets("Tabelle1").Cells(6, 2).Value = "fehlerhafte Zahl"
Exit Sub
End Sub
Public Function fib(n As Integer, stufe As Integer)
Dim ausgabe As String
Dim ausgabewert As String
counter = counter + 1
Start
Beispiel ...
C:\www...
uebung...
uebung6...
uebung5...
Microsof...
15:18


t Excel - uebung5.xls
Datei Bearbeiten Ansicht Einfügen Format Extras Daten Fenster ? Adobe PDF
Frage hier eingeben
ODF importieren ODF exportieren
17.06.2013
Beispiel Zerlegung: Ermittle zu einem Datum den Wochentag
spezifiziere Datum ab 01.01.1900
Datum 17.06.2013
Anzahl Wochentage 41441
Wochentag Montag
berechne
Tabelle1 Tabelle2 Tabelle3
Bereit
Start
Beispiel für d...
C:\www\einf...
uebung6-1....
uebung6-2.xls
uebung5.xls
15:19


t Visual Basic - uebung5.xls
Datei Bearbeiten Ansicht Einfügen Format Debuggen Ausführen Extras Add-Ins Fenster ?
Frage hier eingeben
Projekt - VBAProject
EuroTool (EUROTOOL.XLA)
Sun_ODFP (odfaddin.xla)
VBAProject (uebung5.xls)
DieseArbeitsmappe
Tabelle1 (Tabelle1)
Tabelle2 (Tabelle2)
Tabelle3 (Tabelle3)
VBAProject (uebung6-2.xls)
DieseArbeitsmappe
Modul1
Tabelle1 (Tabelle1)
Tabelle2 (Tabelle2)
Tabelle3 (Tabelle3)
uebung5.xls - Tabelle1 (Code)
(Allgemein)
berechne_Click
Dim tage_monat
tage_monat = Array(31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30,
zaehler = 0
For i = 1900 To (jahr - 1)
zaehler = zaehler + 365
If schaltjahr(i) Then
zaehler = zaehler + 1
End If
Next i
For j = 1 To (monat - 1)
zaehler = zaehler + tage_monat(j - 1)
If j = 2 And schaltjahr(jahr) Then
zaehler = zaehler + 1
End If
Next j
zaehler = zaehler + tag
zahl_tage = zaehler
End Function
Public Function schaltjahr(jahr)
Dim ergebnis As Boolean
If (jahr Mod 4) > 0 Then
ergebnis = False
ElseIf (jahr Mod 100) > 0 Then
ergebnis = True
ElseIf (jahr Mod 400) = 0 Then
ergebnis = True
Else
ergebnis = False
Start
15:19


t Visual Basic - uebung5.xls
uebung5.xls - Tabelle1 (Code)
(Allgemein)
berechne_Click
If (jahr Mod 4) > 0 Then
ergebnis = False
ElseIf (jahr Mod 100) > 0 Then
ergebnis = True
ElseIf (jahr Mod 400) = 0 Then
ergebnis = True
Else
ergebnis = False
End If
schaltjahr = ergebnis
End Function
Public Sub bestimme_tag(anzahl)
wochentag = anzahl Mod 7
Select Case wochentag
Case 0
Worksheets("Tabelle1").Cells(6, 2).Value = "Sonntag"
Case 1
Worksheets("Tabelle1").Cells(6, 2).Value = "Montag"
Case 2
Worksheets("Tabelle1").Cells(6, 2).Value = "Dienstag"
Case 3
Worksheets("Tabelle1").Cells(6, 2).Value = "Mittwoch"
Case 4
Worksheets("Tabelle1").Cells(6, 2).Value = "Donnerstag"
Case 5
Worksheets("Tabelle1").Cells(6, 2).Value = "Freitag"
Case 6
Worksheets("Tabelle1").Cells(6, 2).Value = "Samstag"
End Select
End Sub
Start
15:20


t Visual Basic - uebung5.xls
uebung5.xls - Tabelle1 (Code)
(Allgemein)
berechne_Click
Public Function zahl_tage(tag, monat, jahr) As Long
Dim zaehler As Long
Dim tage_monat
tage_monat = Array(31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30,
zaehler = 0
For i = 1900 To (jahr - 1)
zaehler = zaehler + 365
If schaltjahr(i) Then
zaehler = zaehler + 1
End If
Next i
For j = 1 To (monat - 1)
zaehler = zaehler + tage_monat(j - 1)
If j = 2 And schaltjahr(jahr) Then
zaehler = zaehler + 1
End If
Next j
zaehler = zaehler + tag
zahl_tage = zaehler
End Function
Public Function schaltjahr(jahr)
Dim ergebnis As Boolean
If (jahr Mod 4) > 0 Then
ergebnis = False
ElseIf (jahr Mod 100) > 0 Then
ergebnis = True
ElseIf (jahr Mod 400) = 0 Then
ergebnis = True
Start
15:20


Papierkorb
QuickTime Player
programme.xls
Dienste
RealPlayer
Google Earth
WinVNC
print
VLC media player
GyroTools GO
C:\www\einf-ss13\uebungen
Datei Bearbeiten Ansicht Favoriten Extras ?
Zurück
Suchen
Ordner
Adresse C:\www\einf-ss13\uebungen
Wechseln zu
Ordner
My Music
Program Files
Programme
RECYCLER
src
System Volume Information
temp
WINDOWS
www
einf-ss08
einf-ss09
einf-ss10
einf-ss11
einf-ss12
einf-ss13
atoms
genfiles
png125
png150
png175
uebungen
whiteboard
gbs-ws0809
gbs-ws1112
gbs-ws1213
Name
Größe
Typ
html-seite.html
1 KB
Firefox HTML Docu...
Tücken der Software-Entwickl...
510 KB
Microsoft PowerPoi...
uebung1.doc
28 KB
Microsoft Word-Dok...
uebung1.pdf
12 KB
Adobe Acrobat Doc...
uebung3.xls
20 KB
Microsoft Excel-Arb...
uebung4-1.doc
21 KB
Microsoft Word-Dok...
uebung4-1.pdf
11 KB
Adobe Acrobat Doc...
uebung4-2.xls
22 KB
Microsoft Excel-Arb...
uebung5.xls
38 KB
Microsoft Excel-Arb...
uebung6-1.doc
26 KB
Microsoft Word-Dok...
uebung6-1.pdf
10 KB
Adobe Acrobat Doc...
uebung6-2.xls
46 KB
Microsoft Excel-Arb...
uebung7-1.doc
28 KB
Microsoft Word-Dok...
uebung7-2.xls
21 KB
Microsoft Excel-Arb...
~$bung6-1.doc
1 KB
Microsoft Word-Dok...
Start
Beispiel für den 24.12...
C:\www\einf-ss13\ue...
uebung6-1.doc - Micr...
15:20

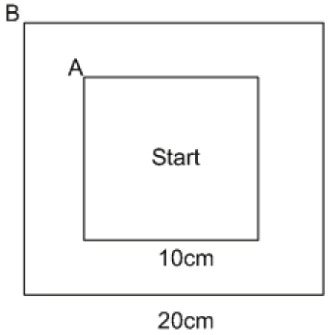
Beispielaufgabe

Zeichnen von zwei konzentrischen Quadraten

Plotterfunktionen

Teilaufgaben

Es existieren die folgenden beiden Teilaufgaben (wobei der Zeichenstift am Start in der Mitte des inneren Quadrats ist).



Position auf A; Quadrat 10 cm zeichnen

Position auf B; Quadrat 20 cm zeichnen

Definition eines Moduls

Nutzung des Moduls (Hauptprogramm)

Plotterfunktionen

Gegeben sind folgende Plotter-Grundfunktionen:

Bewege (x): bewege Zeichenstift um Länge x in aktuelle Zeichenrichtung

Links (x): drehe Zeichenrichtung um x Grad nach links

Rechts (x): drehe Zeichenrichtung um x Grad nach rechts

Stift heben

Stift senken

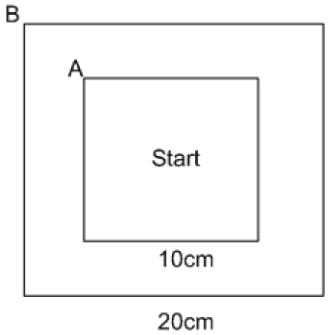
Beispielaufgabe

Zeichnen von zwei konzentrischen Quadraten

Plotterfunktionen

Teilaufgaben

Es existieren die folgenden beiden Teilaufgaben (wobei der Zeichenstift am Start in der Mitte des inneren Quadrats ist).



Position auf A; Quadrat 10 cm zeichnen

Position auf B; Quadrat 20 cm zeichnen

Definition eines Moduls

Nutzung des Moduls (Hauptprogramm)

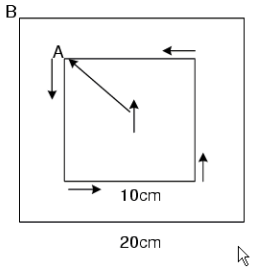
Generated by Targeteam

Definition eines Moduls

Kopf	public Quadratzeichnen(Größe)
Rumpf	Stift senken
	wiederhole 4-mal
	Bewege (Größe)
	Links (90)
	Stift heben

Links (45)
Bewege (10cm*1/2* sqrt(2))
Links (135)
Quadratzeichnen (10cm)
Rechts (135)
Bewege ((20cm-10cm)*1/2*sqrt(2))
Links (135)
Quadratzeichnen (20cm)
Rechts (180)

"sqrt(2)" berechnet die Wurzel von 2.

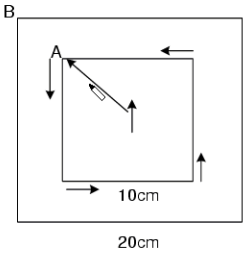


Nutzung des Moduls (Hauptprogramm)

Links (45)
Bewege (10cm*1/2* sqrt(2))
Links (135)
Quadratzeichnen (10cm)
Rechts (135)
Bewege ((20cm-10cm)*1/2*sqrt(2))
Links (135)
Quadratzeichnen (20cm)
Rechts (180)

Name des Moduls
aktueller Wert
, aktueller Wert

"sqrt(2)" berechnet die Wurzel von 2.



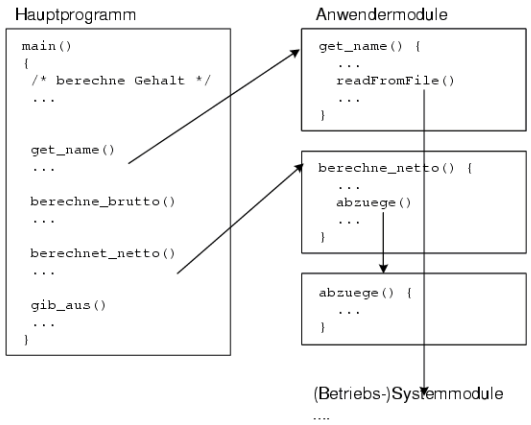
Strukturierung von Algorithmen

Zweck der Strukturierung

- Vereinfachte Darstellung, höhere Lesbarkeit
- Wiederverwendung von Teilen in anderen Kontexten

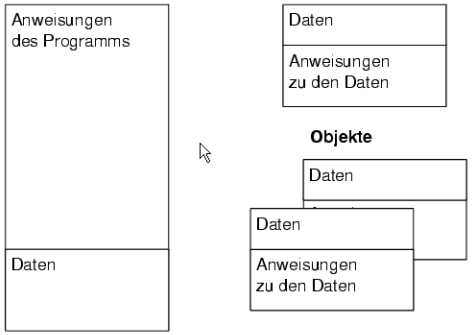
Beispielaufgabe

Hauptprogramm bestehend aus Teilmodulen



Prozedurales / Objektorientieres Programmieren

Prozedurales Programmieren **Objektorientiertes Programmieren**



Anweisungen in einem großen Block (getrennt von den Daten)

Anweisungen jeweils bei den zugehörigen Daten

Prozedurales Programmieren

Sequenz von Anweisungen, ausführen, warten. Basiselemente: Sequenz, Auswahl (if-then-else), Iteration. Trennung von Daten und Anweisungen.

Objektorientiertes Programmieren

Objekte statt Anweisungen. Objekt hat Attribute, Methoden und Ereignisbehandlung. Zusammenfassung von Daten und Anweisungen

Rekursion

Spezielle Form der Modularisierung. Zu definierendes Modul wird in seiner Definition selbst benutzt.

"natürlichere" Darstellung bei bestimmten Algorithmen und Datenstrukturen.

Beispiel Summe

Definiertes Ende einer rekursiven Schachtelung.

```
if (n > 0)
   summe = summe(n-1) + n;
else summe = 0; /* definiertes Ende, wenn n <= 0 */
```

Beispiel 'Fibonacci-Zahlen'

Beispiel 'Größter gemeinsamer Teiler'

Beispiel 'Türme von Hanoi'

Modularisierung von Programmen

Systematischer Aufbau von Programmen durch Zerlegung in eigenständige Einheiten.