31. Bundeswettbewerb Informatik
Aufgabenblatt 1. Runde

Der 31. Bundeswettbewerb Informatik für Jugendliche bis 21 Jahre.

Einsendeschluss ist der 3. Dezember 2012.

Informationen und Unterlagen bitte anfordern bei:
BWINF, Wachsbleiche 7, 53111 Bonn
bwinf@bwinf.de, www.bundeswettbewerb-informatik.de

Grußwort

Informatik ist allgegenwärtig. Sie prägt unseren Alltag in nahezu jeder Hinsicht: Gleich, ob wir telefonieren, Auto fahren, die Mikrowelle oder die Spülmaschine bedienen – ständig hilft sie uns, denn in all diesen Geräten sorgen informationstechnische Systeme für eine intelligente Steuerung.

Die Beschäftigung mit der faszinierenden Welt der Informatik bietet Schülerinnen und Schülern mehrere Chancen: Sie können sich schon früh mit technologischen Zusammenhängen vertraut machen, und sie können eigene Talente entdecken. Eine berufliche Tätigkeit in der Informatik erschöpft sich nicht in der Entwicklung von Computerprogrammen. Sie erlaubt vielmehr einen aktiven Beitrag zum technologischen und wirtschaftlichen Fortschritt.

Der Bundeswettbewerb Informatik und der Jugendwettbewerb „Informatik-Biber“ geben Schülerinnen und Schülern solche wichtigen Einblicke in die Welt der Bits und Bytes. Sie wecken bei jungen Menschen Neugier, Interesse und Begeisterung für das Fach Informatik und tragen außerdem dazu bei, die Bedeutung dieses Faches einer breiteren Öffentlichkeit zu vermitteln.

Über 150.000 Jugendliche haben im vergangenen Jahr durch ihre Teilnahme am Informatik-Biber bewiesen, dass sie keinerlei Berührungsängste mit der Informatik haben. Jene, die durch den Informatik-Biber Lust auf mehr bekommen haben, können ihre Interessen im Rahmen des Bundeswettbewerbs Informatik vertiefen. Beim Lösen der Aufgaben wünsche ich allen Teilnehmerinnen und Teilnehmern gute Ideen und viel Erfolg.

Prof. Dr. Annette Schavan
Bundesministerin für Bildung und Forschung

Die Träger

Gesellschaft für Informatik e.V. (GI)

Die Gesellschaft für Informatik e.V. (GI) ist mit rund 22.000 Mitgliedern die größte Fachgesellschaft der Informatik im deutschsprachigen Raum. Ihre Mitglieder kommen aus allen Sparten der Wissenschaft, aus der Informatikindustrie, aus dem Kreis der Anwender sowie aus Lehre, Forschung, Studium und Ausbildung. In der GI wirken Männer und Frauen am Fortschritt der Informatik mit, im wissenschaftlich-fachlich-praktischen Austausch in etwa 120 verschiedenen Fachgruppen und mehr als 30 Regionalgruppen. Ihr gemeinsames Ziel ist die Förderung der Informatik in Forschung, Lehre und Anwendung, die gegenseitige Unterstützung bei der Arbeit sowie die Weiterbildung. Die GI vertritt hierbei die Interessen der Informatik in Politik und Wirtschaft. Im Web: **www.gi.de**

Fraunhofer-Verbund IuK-Technologie

Als größter europäischer Forschungsverbund für Informations- und Kommunikationstechnik (IuK) versteht sich der Fraunhofer-Verbund IuK-Technologie als Anlaufstelle für Industriekunden auf der Suche nach dem richtigen Ansprechpartner in der anwendungsorientierten IT-Forschung. Die Vernetzung der 4000 Mitarbeiter in bundesweit 19 Instituten ermöglicht die Entwicklung übergreifender branchenspezifischer IT-Lösungen, oft zusammen mit Partnern aus der Industrie, sowie anbieterunabhängige Technologieberatung. Entwickelt werden IuK-Lösungen für die Geschäftsfelder Medizin, Automotive, Produktion, Digitale Medien, E-Business, E-Government, Finanzdienstleister, Sicherheit sowie IT und Kommunikationssysteme.
InnoVisions – Das Zukunftsmagazin des Fraunhofer Verbundes IuK-Technologie informiert Sie über aktuelle Forschungsprojekte auf www.innovisions.de. Weitere Informationen über den Fraunhofer IuK-Verbund gibt es auf www.iuk.fraunhofer.de

Max-Planck-Institut für Informatik

Eine der größten Herausforderungen der Informatik ist die robuste und intelligente Suche nach Information, die grundlegendes Verständnis und automatische Organisation der gewünschten Inhalte voraussetzt. Das Max-Planck-Institut für Informatik widmet sich seit seiner Gründung 1990 diesen Fragestellungen. Das Spektrum der Forschung reicht von allgemeinen Grundlagen der Informatik bis hin zu konkreten Anwendungsszenarien und umfasst Algorithmen und Komplexität, Automatisierung der Logik, Bioinformatik und Angewandte Algorithmik, Computergrafik, Bildverarbeitung und multimodale Sensorverarbeitung sowie Datenbanken und Informationssysteme.
Das Max-Planck-Institut für Informatik unterstützt nachhaltig junge Forscher, die am Institut die Möglichkeit bekommen, ihr eigenes Forschungsgebiet und ihre eigene Gruppe zu entwickeln. Das Institut wirkt seit mehr als 20 Jahren auf Weltklasseniveau durch Publikationen und Software und durch seine jetzigen und ehemaligen Forscher, die Führungsrollen in Wissenschaft und Industrie übernommen haben.
Web: **www.mpi-inf.mpg.de**

GEFÖRDERT VOM

Bundesministerium für Bildung und Forschung

Unter der Schirmherrschaft des Bundespräsidenten

Von der Kultusministerkonferenz empfohlener Schülerwettbewerb

Die Partner

Zusätzlich zur Förderung durch das Bundesministerium für Bildung und Forschung und seine Träger erfährt der Bundeswettbewerb Informatik und damit die Initiative BWINF (Bundesweit Informatiknachwuchs fördern) weitere Unterstützung durch viele Partner. Sie stiften Preise und bieten vor allem spannende Informatik-Workshops für Wettbewerbsteilnehmer an.

Die BWINF-Partner wünschen allen Teilnehmerinnen und Teilnehmern des 31. Bundeswettbewerbs Informatik viel Erfolg!

HPI Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam

max planck institut informatik

INFORMATIK RWTH AACHEN

technische universität dortmund

Google™

O'REILLY®

einstieg informatik **Triff BwInf-Teilnehmer in der Community auf einstieg-informatik.de!**

BwInf.Informatik.erleben

Bundeswettbewerb Informatik

Der Bundeswettbewerb Informatik (BwInf) wurde 1980 von der Gesellschaft für Informatik e.V. (GI) auf Initiative von Prof. Dr. Volker Claus ins Leben gerufen. Ziel des Wettbewerbs ist, Interesse an der Informatik zu wecken und zu intensiver Beschäftigung mit ihren Inhalten und Methoden sowie den Perspektiven ihrer Anwendung anzuregen. Er gehört zu den bundesweiten Schülerwettbewerben, die von den Kultusministerien der Länder empfohlen werden. Gefördert wird er vom Bundesministerium für Bildung und Forschung und steht unter der Schirmherrschaft des Bundespräsidenten. Der BwInf ist Kern von „Bundesweit Informatiknachwuchs fördern“ (BWINF), einer gemeinsamen Initiative von GI, Fraunhofer-Verbund IuK-Technologie und Max-Planck-Institut für Informatik. Die Gestaltung des Wettbewerbs und die Auswahl der Sieger obliegen dem Beirat; Vorsitzende: Prof. Dr. Nicole Schweikardt, Universität Frankfurt. Die Auswahl und Entwicklung von Aufgaben und die Festlegung von Bewertungsverfahren übernimmt der Aufgabenausschuss; Vorsitzender: Prof. Dr. Peter Rossmanith, RWTH Aachen. Die Geschäftsstelle des Wettbewerbs mit Sitz in Bonn ist für die fachliche und organisatorische Durchführung zuständig; Geschäftsführer: Dr. Wolfgang Pohl.

Drei Runden

Der Wettbewerb beginnt jedes Jahr im September, dauert etwa ein Jahr und besteht aus drei Runden. In der ersten und zweiten Runde sind die Wettbewerbsaufgaben zu Hause selbstständig zu bearbeiten. Dabei können die Aufgaben der ersten Runde mit guten grundlegenden Informatikkenntnissen gelöst werden; die Aufgaben der zweiten Runde sind deutlich schwieriger. In der ersten Runde ist Gruppenarbeit zugelassen und erwünscht. In der zweiten Runde ist dann eigenständige Einzelarbeit gefordert; die Bewertung erfolgt durch eine relative Platzierung der Arbeiten. Die bis zu dreißig bundesweit Besten der zweiten Runde werden zur dritten Runde, einem Kolloquium, eingeladen. Darin führt jeder Gespräche mit Informatikern aus Schule und Hochschule und analysiert und bearbeitet im Team zwei Informatik-Probleme.

Juniorliga

Um die Teilnahme jüngerer Schülerinnen und Schüler am BwInf zu fördern, gibt es in der ersten Runde die Juniorliga. Für Schüler, die nicht älter als 16 Jahre sind, werden zwei leichtere Aufgaben gestellt, die Junioraufgaben. Stammt eine Einsendung von Schülern, die alle die Altersgrenze für Junioraufgaben erfüllen, und sind darin nur Junioraufgaben bearbeitet, nimmt sie in der Juniorliga teil. Die Juniorliga wird getrennt bewertet, Preise werden separat vergeben.

Die Chancen

Preise

In allen Runden des Wettbewerbs wird die Teilnahme durch eine Urkunde bestätigt. In der ersten Runde werden auf den Urkunden erste und zweite Preise sowie Anerkennungen unterschieden; mit einem Preis ist die Qualifikation für die zweite Runde verbunden. In der zweiten Runde gibt es erste, zweite und dritte Preise; jüngere Teilnehmer haben die Chance auf eine Einladung zu einer Schülerakademie. Ausgewählte Gewinner eines zweiten Preises erhalten einen Buchpreis des Verlags O'Reilly; erste Preisträger werden zur dritten Runde eingeladen, die im Herbst 2013 in Kaiserslautern ausgerichtet wird, vom Fraunhofer IESE und der TU Kaiserslautern.

Die dort ermittelten Bundessieger werden in der Regel ohne weiteres Aufnahmeverfahren in die Studienstiftung des deutschen Volkes aufgenommen. Zusätzlich sind für den Bundessieg, aber auch für andere besondere Leistungen Geld- und Sachpreise vorgesehen.

Informatik-Olympiade

Ausgewählte Teilnehmerinnen und Teilnehmer können sich in mehreren Trainingsrunden für das vierköpfige deutsche Team qualifizieren, das an der Internationalen Informatik-Olympiade 2014 in Taiwan teilnimmt.

Informatik-Workshops etc.

Informatik-Workshops exklusiv für TeilnehmerInnen werden in Baden-Württemberg, vom Hasso-Plattner-Institut, von Hochschulen wie der RWTH Aachen oder der TU Dortmund und dem Max-Planck-Institut für Informatik (2. Runde) veranstaltet. Bei einigen von Fraunhofer-Instituten veranstalteten „Talent Schools" gibt es reservierte BwInf-Plätze. Die Firma Google lädt ausgewählte Teilnehmerinnen zum „Girls@Google Day" und einige der Allerbesten zu einem Treffen mit Informatik-Talenten aus ganz Europa ein.

Ausgewählte Endrundenteilnehmer werden im Herbst 2013 vom Bundesministerium für Bildung und Forschung zum „Tag der Talente" eingeladen.

Eine Einsendung zur zweiten Runde kann in vielen Bundesländern als besondere Lernleistung in die Abiturwertung eingebracht werden.

Preise für BwInf-Schulen

Für eine substanzielle Beteiligung am Wettbewerb werden Schulpreise vergeben: An mindestens 3 vollwertigen Einsendungen (also mit je mindestens 3 bearbeiteten Aufgaben) zur 1. Runde – wobei eine vollwertige Einsendung durch zwei Einsendungen in der Juniorliga ersetzt werden kann – müssen mindestens 10 Schülerinnen und Schüler einer Schule, darunter bei gemischten Schulen mindestens 2 Jungen und mindestens 2 Mädchen, beteiligt sein. Schulen, die diese Bedingung erfüllen, werden als „BwInf-Schule 2012/2013" ausgezeichnet: sie erhalten ein entsprechendes Zertifikat, ein Label zur Nutzung auf der Schul-Website und einen Gutschein im Wert von **300 Euro** für Bücher oder andere für den Informatikunterricht benötigte Dinge.

Die Regeln

Teilnahmeberechtigt

… sind Jugendliche, die nach dem 3.12.1990 geboren wurden. Sie dürfen jedoch zum 1.9.2012 noch nicht ihre (informatikbezogene) Ausbildung abgeschlossen oder eine Berufstätigkeit begonnen haben. Personen, die im Wintersemester 2012/13 an einer Hochschule studieren, sind ausgeschlossen, falls sie nicht gleichzeitig noch die Schule besuchen. Jugendliche, die nicht deutsche Staatsangehörige sind, müssen wenigstens vom 1.9. bis 3.12.2012 ihren Wohnsitz in Deutschland haben oder eine staatlich anerkannte deutsche Schule im Ausland besuchen.

Junioraufgaben dürfen von bis zu 16-Jährigen bearbeitet werden (geboren nach dem 3.12.1995) bzw. von Gruppen mit mindestens einem solchen Mitglied.

Weiterkommen

An der zweiten Runde dürfen jene teilnehmen, die allein oder mit ihrer Gruppe wenigstens drei Aufgaben der ersten Runde weitgehend richtig gelöst haben. Für die dritte Runde qualifizieren sich die besten ca. 30 Teilnehmer der zweiten Runde.

In der Juniorliga gibt es noch keine zweite Runde.

Einsendungen

… enthalten Bearbeitungen zu mindestens einer Aufgabe und werden von Einzelpersonen oder Gruppen abgegeben. Eine Einsendung besteht aus einem schriftlichen und einem elektronischen Teil. Zu jeder bearbeiteten Aufgabe enthält der

schriftliche Teil (Dokumentation)
eine Beschreibung der Lösungsidee und Beispiele, die die Korrektheit der Lösung belegen. Ist ein Programm gefordert, sollen außerdem die Umsetzung der Lösungsidee in das Programm erläutert und die wichtigsten Teile des Quelltextes hinzugefügt werden. Die Dokumentation muss ausgedruckt abgegeben werden. Achtung: eine gute Dokumentation muss nicht lang sein! Der

elektronische Teil
enthält für Aufgaben, in denen ein Programm gefordert ist, das lauffähige Programm selbst und den kompletten Quelltext des Programms. Er muss, zusammen mit einer elektronischen Fassung der Dokumentation, auf CD/DVD abgegeben werden.

Anmeldung

Die Anmeldung ist bis zum Einsendeschluss möglich, und zwar online unter
www.bundeswettbewerb-informatik.de/anmeldung
In diesem Jahr ist zum ersten Mal eine **Lehreranmeldung** möglich. Bitte erkundigen Sie sich unter bwinf@bwinf.de nach den Modalitäten.

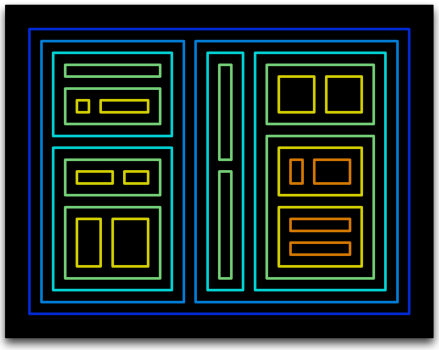
Einsendung an: BWINF, Wachsbleiche 7, 53111 Bonn

Einsendeschluss: 3.12.2012 (Datum des Poststempels)

Verspätete Einsendungen können nicht berücksichtigt werden. Der Rechtsweg ist ausgeschlossen. Die Einsendungen werden nicht zurückgegeben. Der Veranstalter erhält das Recht, die Beiträge in geeigneter Form zu veröffentlichen.

Beispiellösung: Rechteckschoner

Als eine ältere Mitschülerin von den letzten Zweitrundenaufgaben erzählte, hatte Moni die Idee zu einem eigenen Bildschirmschoner: Gezeichnet werden soll ein Rechteck, das zwei Rechtecke enthält, die jeweils wieder zwei Rechtecke enthalten usw. Damit das nett aussieht, soll bei der Rechteckschachtelung der Zufall eine sinnvolle Rolle spielen, und ein enthaltenes Rechteck soll sich nicht nur in der Größe von dem es unmittelbar enthaltenden Rechteck unterscheiden – etwa so:



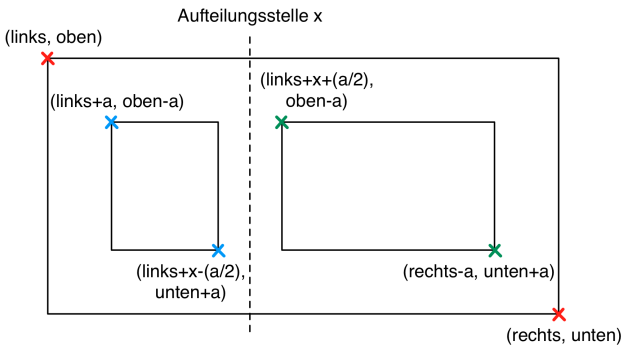
Nach dem Aufbau soll das fertige Rechteckschachtelbild immer wieder so aktualisiert werden, dass mal das eine, mal das andere Rechteck (evtl. mit allen enthaltenen Rechtecken) sein Aussehen verändert.

Aufgabe

Entwickle einen solchen „Rechteckschoner“. Demonstriere seine Funktion an mindestens drei verschiedenen Rechteckschachtelbildern mit jeweils zwei Veränderungen.

Lösungsidee

Um ein einzelnes Rechteck zeichnen zu können, müssen seine Position und seine Farbe festgelegt werden. Die Farbe wird zufällig aus einer festen Liste gewählt. Die Position eines Rechtecks wird durch zwei diagonal gegenüberliegende Eckpunkte festgelegt. Beim äußersten Rechteck werden die Eckpunkte so bestimmt, dass sie alle den gleichen Abstand zu den Bildschirmrändern haben. Um zwei Rechtecke in das äußere zu zeichnen, wird dessen Fläche zufällig aufgeteilt. Die Eckpunkte der inneren Rechtecke werden so bestimmt, dass sie den gleichen Abstand zueinander haben wie zum äußeren Rechteck. Die folgende Abbildung zeigt, wie die Eckpunkte der inneren Rechtecke aus den Eckpunkten des äußeren Rechtecks (links, oben) und (rechts, unten) sowie der „Aufteilungsstelle“ x berechnet werden. Das äußere Rechteck wird senkrecht aufgeteilt, wenn es breiter ist als hoch (wie in der Abbildung); andernfalls wird es waagerecht aufgeteilt, und die Eckpunkte der inneren Rechtecke werden entsprechend berechnet.



Damit das Rechteckbild gleichmäßig gefüllt wird, wird eine Liste der noch zu zeichnenden Rechtecke geführt. Diese Liste wird nach und nach abgearbeitet. Beim Füllen eines Rechtecks werden die beiden inneren Rechtecke hinten an die Liste angehängt. So wird erreicht, dass die Rechtecke Stufe für Stufe gezeichnet werden: erst das äußere Rechteck, dann dessen zwei innere Rechtecke, dann deren vier innere Rechtecke usw. Da ein Rechteck nur dann gefüllt wird, wenn es noch genügend Platz für zwei innere Rechtecke hat, endet dieser Prozess, wenn alle zu zeichnenden Rechtecke die Mindestgröße unterschreiten.

Alle gezeichneten Rechtecke werden in einer Liste abgespeichert. Ist das Bild einmal fertig, wird aus dieser Liste zufällig ein Rechteck ausgewählt. Es wird dann mit seinem Inhalt zunächst komplett gelöscht und übermalt und dann mit neuen inneren Rechtecken neu gezeichnet – auf die gleiche Art und Weise wie ursprünglich das äußerste Rechteck.

Umsetzung

Die Lösungsidee wird in der Sprache Python umgesetzt, und zwar mit Hilfe des Moduls „turtle“, das viele Funktionen zum Zeichnen von Figuren bereitstellt.

Alle Eigenschaften und wichtigen Funktionen eines Rechtecks werden in einer Klasse `Rechteck` zusammengefasst. Eigenschaften sind im Wesentlichen die Koordinaten der Eckpunkte (`links`, `oben`, `rechts` und `unten`) und die Farbe `farbe`. Als Funktionen (oder besser: Methoden) eines Rechtecks werden definiert:

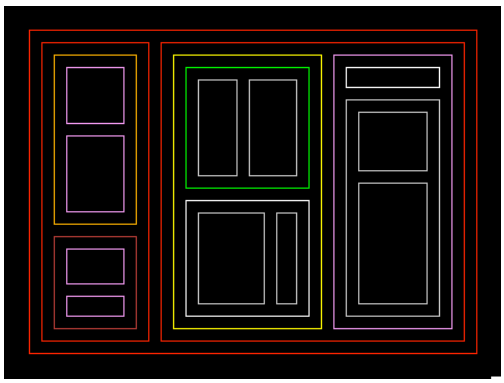
- `zeichne` : nutzt Funktionen des Moduls turtle, um anhand der Eckpunkte ein Rechteck der gesetzten Farbe zu zeichnen.
- `fuelle` : bestimmt, ob das Rechteck groß genug ist, um zwei innere Rechtecke zu erhalten. Wenn ja, wird das Rechteck aufgeteilt und es werden zwei innere Rechtecke erzeugt.
- `loesche` : entfernt das Rechteck aus der Liste aller Rechtecke und löscht rekursiv seine Inhaltsrechtecke; jedes gelöschte Rechteck wird abschließend in Hintergrundfarbe neu gezeichnet und damit auch aus der Anzeige gelöscht.

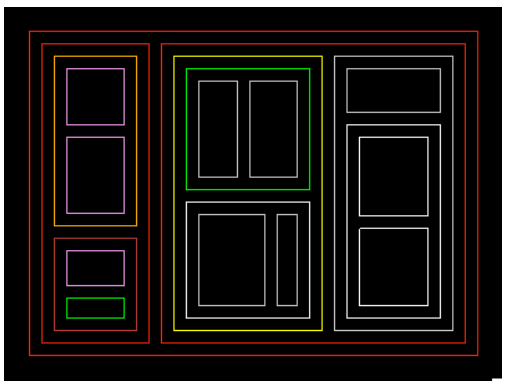
Alle Rechtecke (also alle Objekte der Klasse `Rechteck`) werden in der globalen Liste `alle_rechtecke` gespeichert.

Die zentrale Funktion zum Zeichnen eines Rechtecks ist `zeichne_rechtecke`; sie benutzt die lokale Liste `aktuelle_rechtecke`, um – wie in der Lösungsidee beschrieben – die Rechtecke stufenweise zu erzeugen und zu zeichnen.

Beispiel

Aus Platzgründen zeigen wir nur ein Rechteckschachtelbild mit einer Veränderung. Die Bilder sind 400 mal 300 Pixel groß. Die Veränderungen vom linken zum rechten Bild sind: Das innerste Rechteck links unten wurde gelöscht und in anderer Farbe neu gezeichnet; außerdem wurde im rechten Drittel des Bildes das violette Rechteck gelöscht und mit neuen inneren Rechtecken neu gezeichnet.





Quelltext

```
# Modulimport
import turtle
import random

# Hier kann die Größe des Fensters an den Bildschirm angepasst werden.
SCREEN_WIDTH = 400
SCREEN_HEIGHT = 300

SPEED = 5 # Zeichengeschwindigkeit

# Farben für die Rechtecke
COLORS = ["red", "white", "yellow", "green", "blue",
          "orange", "brown", "pink", "gray", "violet"]
BGCOLOR = "black" # Hintergrundfarbe

# Rechteck-Klasse mit Eigenschaften und Operationen
class Rechteck:

    def __init__(self,links,oben,rechts,unten):
        '''Die Eigenschaften des Rechtecks werden initialisiert.'''
        self.links=links
        self.oben=oben
        self.rechts=rechts
        self.unten=unten
        self.hoehe=oben-unten # nicht nötig, aber nützlich
        self.breite=rechts-links # nicht nötig, aber nützlich
        self.farbe=self.waehle_farbe()
        self.inhalt=[] # Liste für die beiden direkt enthaltenen Rechtecke

    def waehle_farbe(self):
        '''Die Linienfarbe wird zufällig ausgewählt.'''
        random.shuffle(COLORS)
        stift.color(COLORS[0])

    def zeichne(self):
        '''Das Rechteck wird mit Turtle-Kommandos gezeichnet.'''
        stift.up()
        stift.goto(self.links,self.oben)
        stift.down()
        stift.fd(self.breite)
        stift.right(90)
        stift.fd(self.hoehe)
        stift.right(90)
        stift.fd(self.breite)
        stift.right(90)
        stift.fd(self.hoehe)
        stift.right(90)

    def fuelle(self):
        '''Das Rechteck bekommt zwei Rechtecke als Inhalt, wenn es groß
        genug ist. Wenn es breiter als hoch ist, werden die enthaltenen
        Rechtecke nebeneinander, sonst übereinander platziert.'''
        if self.breite >= 60 and self.hoehe >= 60:
            if self.breite > self.hoehe:
                x = random.uniform(self.links+30, self.rechts-30)
                r1 = Rechteck(self.links+10, self.oben-10, x-5, self.unten+10)
                r2 = Rechteck(x+5, self.oben-10, self.rechts-10, self.unten+10)
                self.inhalt = [r1,r2]
            else:
                x = random.uniform(self.oben-30,self.unten+30)
                r1 = Rechteck(self.links+10, self.oben-10, self.rechts-10, x+5)
                r2 = Rechteck(self.links+10, x-5, self.rechts-10, self.unten+10)
                self.inhalt = [r1,r2]
        return self.inhalt
```

```
    def loesche(self):
        '''Das Rechteck wird mit seinem ganzen Inhalt, also rekursiv,
        aus der Liste mit allen Rechtecken gelöscht. Die Rechtecke werden
        von innen nach außen mit der Hintergrundfarbe übermalt.'''
        global alle_rechtecke
        alle_rechtecke.remove(self)
        for irechteck in self.inhalt:
            irechteck.loesche()
        stift.color(BGCOLOR)
        self.zeichne()


def schluss(x,y):
    '''Das Turtle-Fenster wird geschlossen.'''
    global screen
    screen.bye()

def setup_turtle():
    '''Turtle(-Stift) und Turtle-Fenster werden vorbereitet.'''
    global stift, screen
    stift=turtle.Turtle()
    screen=stift.getscreen()
    screen.title('Rechteckschoner')
    screen.bgcolor(BGCOLOR)
    screen.setup(SCREEN_WIDTH,SCREEN_HEIGHT,None,None)
    screen.listen()
    screen.onclick(schluss) # Mit einem Klick die Turtle beenden.
    stift.reset()
    stift.speed(SPEED)

def zeichne_rechtecke(ausseneck):
    '''Ein Rechteck wird mit seinen inneren Rechtecken gezeichnet.'''
    aktuelle_rechtecke = []
    aktuelle_rechtecke.append(ausseneck)
    # Alle Rechtecke werden gezeichnet, so lange Platz da ist.
    for r0 in aktuelle_rechtecke:
        for r in r0.fuelle():
            aktuelle_rechtecke.append(r)
        r0.zeichne()
    return aktuelle_rechtecke

def main():
    global alle_rechtecke
    setup_turtle()
    alle_rechtecke = []
    # Mit dem ersten, äußersten Rechteck geht es los.
    starteck = Rechteck(-SCREEN_WIDTH/2+20,SCREEN_HEIGHT/2-20,
                        SCREEN_WIDTH/2-20,-SCREEN_HEIGHT/2+20)
    # Endlosschleife, in der Rechtecke gezeichnet und gelöscht werden.
    while 1:
        alle_rechtecke.extend(zeichne_rechtecke(starteck))
        # Ein Rechteck zum Löschen auswählen und löschen ...
        wegeck = random.choice(alle_rechtecke)
        wegeck.loesche()
        # ... und mit neuer Farbe zum neuen Starteck machen.
        wegeck.waehle_farbe()
        starteck = wegeck
    return "EVENTLOOP"

if __name__ == '__main__':
    msg = main()
    print(msg)
    mainloop()
```

Skyline

Das Wahrzeichen von Boomtown ist die Needle, ein schlanker, 100m hoher Turm. Nach ihrem Bau wurde einst beschlossen, dass kein Gebäude in Boomtown höher als die Needle sein darf. Nun sollen viele neue Gebäude in Boomtown errichtet werden, und die Bauherren wollen höher hinaus.

Der Rat von Boomtown überlegt, dass Gebäude, die weiter von der Needle weg sind, höher sein dürfen. So können die Bauherren befriedigt werden, und gleichzeitig bleibt die Needle als markantes Wahrzeichen der Stadt erhalten. Der Rat erlässt also die Vorschrift, dass mit jeden 100m Abstand von der Needle 1m höher gebaut werden darf. Also ist die maximal erlaubte Höhe bei einem Abstand von 0m bis 99m: 100m, bei einem Abstand von 100m bis 199m: 101m usw.

Nun soll für die vielen Bauanfragen entschieden werden, wie hoch das Gebäude jeweils sein darf. Zum Glück hatten die Stadtväter von Boomtown einst ein Koordinatensystem mit der Einheitslänge 1m eingeführt, in dem die Needle die Koordinaten (0,0) hat. Für die Grundrisse aller Gebäude gelten folgende Regeln:

- sie sind rechteckig;
- ihre Eckpunkte haben ganzzahlige Koordinaten;
- ihre Seiten liegen parallel zu den Achsen des Koordinatensystems.

Das Bild unten zeigt drei Gebäude mit den Grundrissen (angegeben durch die Eckpunkte)

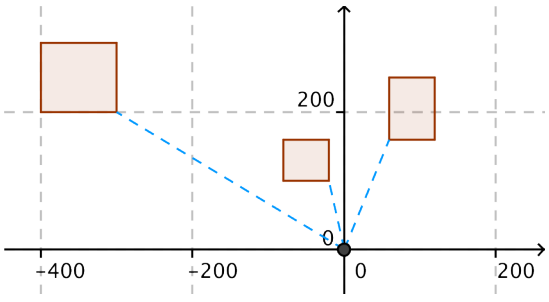
(−80, 100); (−20, 100); (−20, 160); (−80, 160)
(60, 160); (120, 160); (120, 250); (60, 250)
(–400,200); (–300, 200); (–300, 300); (–400, 300)

Junioraufgabe 1

Schreibe ein Programm, das für eine Liste von Grundrissen die erlaubten Höhen der Gebäude berechnet.

Für das Beispiel im Bild lautet die richtige Ausgabe:

101
101
103



Verben

Im Deutschunterricht wird gerade die Konjugation deutscher Verben behandelt. Mehmet denkt sich dabei: „Wozu braucht man da überhaupt einen Deutschlehrer? Das sind doch ganz einfache Regeln. Die programmiere ich mal fix; dann kann mein Smartphone hoffentlich die Hausaufgaben der nächsten Wochen erledigen.“ Nach einiger Zeit muss er aber zugeben, dass sein Programm immer noch nicht fertig ist und dass er nach fast jeder Deutschstunde irgendwelche kniffligen Fälle mit seinem Lehrer bespricht.

Junioraufgabe 2

Betrachte einen kleinen Teil von dem, was Mehmet vorhat: Schreibe ein Programm, das aus einer beliebigen Form eines deutschen Verbs durch Anwendung von Regeln versucht, die Grundform (den Infinitiv) des Verbs herauszufinden. Erhält das Programm z. B. die Eingabe „sagst“, soll es „sagen“ ausgeben.

Dein Programm muss zumindest bei Eingaben wie den folgenden die richtige Ausgabe produzieren:

sagst
leitete
geforscht
schweigend
trag

Zeige durch geeignete Eingabe-Ausgabepaare, was dein Programm kann. Zeige auch durch andere (vielleicht lustige) Beispiele, was es nicht kann.

Freiwillige Zusatzaufgabe:

Wenn du Lust hast, kannst du probieren, dein Programm so weiter zu entwickeln, dass es sich auch von hinterhältigen Eingaben wie „gebet“, „arbeite“, „genehmigt“, „gehört“ oder „mitgehalten“ / „mitgestalten“ nicht aus der Fassung bringen lässt. Das ist aber schwierig; und unregelmäßige Verben muss dein Programm sowieso nicht korrekt behandeln.

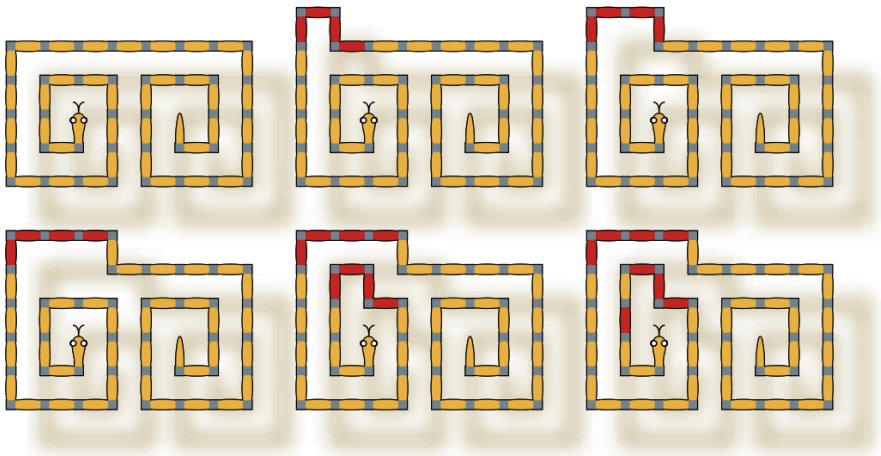
Frühsport

Die BWINF-Schlange Digitail wacht nach einer durchrechneten Nacht mit heißen Algorhythmen und zuviel Soft Ware ganz verknäult auf und möchte wieder ihr altes geradliniges Selbst werden. „Wie gut, dass ich in einer zwei- und nicht dreidimensionalen Welt lebe“, denkt sie, „denn sonst könnte ich echt ein Problem haben“. Auch so braucht sie aber deine Hilfe, sich zu entknäulen.

Digitail besteht aus geraden Ringeln der Länge 1, die Gitterpunkte mit ganzzahligen Koordinaten verbinden. Sie ist normalerweise goldfarben; aber sie kann sich vorübergehend strecken und wird dann stellenweise rot: Aus einem oder zwei aufeinander folgenden goldenen Ringeln können doppelt so viele rote Ringel werden. Wenn sich die Schlange wieder zusammenzieht, werden aus zwei oder vier aufeinander folgenden roten Ringeln halb so viele goldene Ringel. Außerdem kann die Schlange ein oder zwei aufeinander folgende Ringel beliebiger Farbe verschieben, und zwei aufeinander folgende Ringel können ihre Farben tauschen. Diese akrobatischen Übungen sind allerdings alle nur möglich, wenn die dafür benötigten Gitterpunkte vorher frei waren und die Schlange zusammenhängend bleibt.

Aufgabe 1

Schreibe ein Programm, das eine beliebig verknäulte anfangs goldene Schlange einliest und eine Folge von Schritten der beschriebenen Art berechnet, nach deren Ausführung alle Ringel der Schlange auf einer Geraden liegen und (wieder) golden sind. Visualisiere die Ausführung der Schritte so, dass man sehen kann, wie sich die Schlange entknäult.



Geldtransporter

Zum Transport von Geld werden schwer bewachte Geldtransporter eingesetzt. In einem solchen Geldtransporter können Koffer mit Münzen transportiert werden. Die Koffer enthalten Münzen in unterschiedlicher Menge und mit unterschiedlichem Gesamtwert. Entsprechend unterscheiden sich die Koffer in Gewicht und Wert.

Da kommt so einiges an Gewicht zusammen. Es ist daher notwendig, den Transporter gleichmäßig zu beladen, so dass er nicht zu einer Seite umkippen kann.

Unser Transporter hat links und rechts je einen Kofferraum. Für jeden Kofferraum lassen sich Gesamtwert und Gesamtgewicht der darin enthaltenen Koffer bestimmen. Damit der Transporter keine Schlagseite bekommt, müssen die Koffer so eingeräumt werden, dass die Differenz zwischen den beiden Gesamtgewichten minimal ist. Aus versicherungstechnischen Gründen dürfen die beiden Gesamtwerte sich außerdem um höchstens 10.000 Euro unterscheiden.

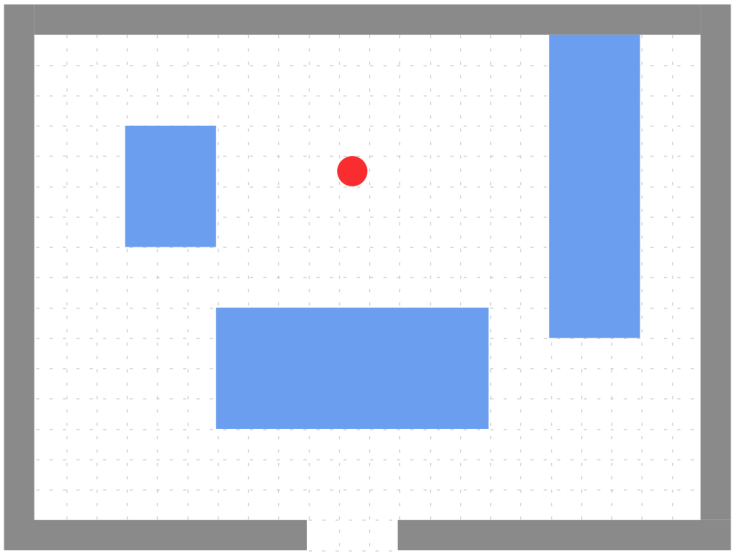
Aufgabe 2

Schreibe ein Programm zur Verteilung der Geldkoffer auf die beiden Kofferräume. Überprüfe dein Programm mit den auf www.bundeswettbewerb-informatik.de abgelegten Beispielen mit Angaben zu Werten und Gewichten der einzelnen Koffer.

Turn90

Der Robotertyp „Turn90“ ist in seiner Bewegungsfähigkeit eingeschränkt: Er kann sich nur vorwärts bewegen und um 90 Grad drehen. Außerdem hat er keinerlei visuelle Sensoren, kann also nicht „sehen“. Seine Kontaktsensoren und auch seine Rechenfähigkeiten sind aber sehr ordentlich.

Ein neuer Auftrag zur Programmierung von Turn90-Robotern scheint darauf Rücksicht zu nehmen: Der Roboter soll befähigt werden, in einem Raum mit Hindernissen immer den einzigen Ausgang zu finden. Die Programmierer sind erleichtert, als sie hören, dass dem Raum ein Raster aus Quadraten zugrunde liegt. Die Wände füllen die äußeren Quadrate des Rasters, die Hindernisse füllen rechteckige Gruppen von Rasterquadraten aus. Der Roboter befindet sich zu Beginn in einem bestimmten Quadrat. Der Ausgang ist eine mindestens ein Quadrat breite Lücke in einer Raumwand. Das Bild unten zeigt einen Beispielraum mit Wänden und Hindernissen. Der Roboter ist als Kreis eingezeichnet.

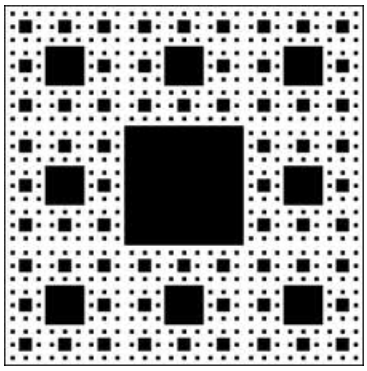


Aufgabe 3

Schreibe ein Programm, mit dem ein Turn90-Roboter unter den genannten Bedingungen immer den Ausgang findet. Du kannst davon ausgehen, dass die Räume so gestaltet sind, dass es einen Weg vom Roboter zum Ausgang gibt.

Schicke uns für die drei Räume, die auf den BwInf-Webseiten angegeben sind, und für zwei weitere Räume je ein grafisches Protokoll des Roboterweges. Lasse entweder dein Programm den Weg zeichnen, oder stelle ein schriftliches Protokoll des Weges (das du dann bitte mitschickst) selbst grafisch dar.

SVG-Fraktale



Ein Fraktal ist eine selbstähnliche Figur. Es entsteht, indem, ausgehend von einer Grundfigur, geometrische Objekte schrittweise verfeinert werden. Ein bekanntes Fraktal ist zum Beispiel der Sierpinski-Teppich (s. Bild).

Scalable Vector Graphics (SVG) ist eine Sprache, in der sich geometrische Objekte mit leicht lesbaren Sprachelementen beschreiben lassen, um sie z. B. in Webbrowsern anzuzeigen. Eine Definition findet man unter http://www.w3.org/TR/SVG11/. Für eine Lösung dieser Aufgabe genügen aber der „SVG-Rahmen“:

```
<?xml version="1.0"?>
<svg xmlns="http://www.w3.org/2000/svg"> elemente </svg>
```

und die folgenden Elemente:

```
<rect x="zahl" y="zahl" width="zahl" height="zahl" fill="farbe"/>
```

```
<circle cx="zahl" cy="zahl" r="zahl" fill="farbe"/>
```

```
<line x1="zahl" y1="zahl" x2="zahl" y2="zahl" stroke="farbe" stroke-width="zahl"/>
```

Die Grundfigur (erste Stufe) des Sierpinski-Teppichs, ein einfaches weißes Quadrat, wird in einer SVG-Datei so beschrieben:

```
<?xml version="1.0"?>
<svg xmlns="http://www.w3.org/2000/svg">
  <rect x="0" y="0" width="180" height="180" fill="white"/>
</svg>
```

Ein Verfeinerungsschritt besteht generell daraus, ein Objekt durch ein oder mehrere neue Objekte zu ersetzen. Beim Sierpinski-Teppich wird jedes weiße Quadrat durch neun kleinere, gleich große Quadrate ersetzt, von denen das mittlere schwarz und alle anderen weiß sind. Die zweite Stufe des Sierpinski-Teppichs wird also in einer SVG-Datei so beschrieben:

```
<?xml version="1.0"?>
<svg xmlns="http://www.w3.org/2000/svg">
  <rect x="0" y="0" width="60" height="60" fill="white"/>
  <rect x="60" y="0" width="60" height="60" fill="white"/>
  <rect x="120" y="0" width="60" height="60" fill="white"/>
  <rect x="0" y="60" width="60" height="60" fill="white"/>
  <rect x="60" y="60" width="60" height="60" fill="black"/>
  <rect x="120" y="60" width="60" height="60" fill="white"/>
  <rect x="0" y="120" width="60" height="60" fill="white"/>
  <rect x="60" y="120" width="60" height="60" fill="white"/>
  <rect x="120" y="120" width="60" height="60" fill="white"/>
</svg>
```

Das Bild oben zeigt die fünfte Stufe des Sierpinski-Teppichs.

Aufgabe 4

Wähle eine der auf www.bundeswettbewerb-informatik.de bereit gestellten Grundfiguren oder denke dir selbst eine Grundfigur aus. Schreibe ein Programm, das die Grundfigur oder eine bereits berechnete Verfeinerungsstufe aus einer SVG-Datei einliest und eine neue SVG-Datei erzeugt, die die nächste Stufe beschreibt. Die SVG-Dateien sollen von gängigen Web-Browsern (z. B. Firefox) angezeigt werden können.

Als Programmiersprache zur Verarbeitung von XML-Dialekten wie SVG eignet sich XSLT, aber auch andere Sprachen sind zugelassen. Es genügt, wenn eine Ausführung des Programms nur einen Verfeinerungsschritt vornimmt.

Kaffeerunde

Die Inhaberin der Firma Nash GmbH wünscht sich glückliche Mitarbeiter und möchte ihnen fürderhin kostenlos Kaffee anbieten. Hierzu wird Kaffee zur Verfügung gestellt sowie eine Kaffeemaschine mit einer Kanne. Die Mitarbeiter müssen die Maschine selbst bedienen (nach einer Einführungszeit von zehn Tagen, in denen ein Assistent der Inhaberin immer für eine volle Kanne sorgt). Abends wird die Kanne stets entleert.

Die Kanne fasst zehn Tassen und es gibt 15 Mitarbeiter. Jeder Mitarbeiter möchte gern drei Tassen täglich trinken, und zwar je eine vormittags (8–10 Uhr), mittags (12–14 Uhr) und nachmittags (15–17 Uhr). In jedem dieser zweistündigen Intervalle erscheinen die Kaffeetrinker in einer zufälligen Reihenfolge in der Kaffeeküche. Sie verhalten sich dann so:

Falls sich noch Kaffee in der Kanne befindet, nehmen sie sich eine Tasse daraus und gehen wieder. Ist die Kanne aber leer, dann hängt ihr weiteres Verhalten davon ab, ob sie glücklich sind. Falls ja, kochen sie eine neue volle Kanne und nehmen sich davon selbst eine Tasse. Sind sie aber nicht glücklich, dann haben sie keine Lust, für andere Kaffee mitzukochen, und gehen unverrichteter Dinge wieder weg.

Ob eine Person an einem bestimmten Tag glücklich ist, entscheidet sich direkt morgens nach dem Aufstehen und hängt davon ab, wie oft sie in den letzten zehn Tagen selbst Kaffee kochte und wie viele Tassen Kaffee sie in diesem Zeitraum trank. Die genaue Definition lautet:

> Falls eine Person in den letzten zehn Tagen n Tassen trank und m-mal Kaffee kochte, dann ist sie genau dann glücklich, wenn $n \geq 10$ und $m/n < \alpha$ ist, wobei α ein persönlicher Schwellenwert ist.

Aufgabe 5

Schreibe ein Programm, das die Kaffeetrinker simuliert, wobei α der Einfachheit halber für alle identisch sein möge. Wie viele Tassen trinkt ein Mitarbeiter täglich im Durchschnitt auf lange Sicht, abhängig von verschiedenen Schwellenwerten α?

Ist das Ergebnis so, wie du es erwartetest?

Teilnehmen

Einsendeschluss ist der 3. Dezember 2012 (Datum des Poststempels)

Einsendung an: BWINF, Wachsbleiche 7, 53111 Bonn

Fragen zu den Aufgaben?

per Telefon: **0228 378646** zu üblichen Bürozeiten; per E-Mail: **bwinf@bwinf.de**

Diskutiere über die Aufgaben mit den Mitgliedern der **EI Community**:
www.einstieg-informatik.de

Anmeldung

online unter www.bundeswettbewerb-informatik.de/anmeldung

Hinweise zur Einsendung

Für jede bearbeitete Aufgabe solltest du im schriftlichen Teil deiner Einsendung (der **Dokumentation**)
- deine **Lösungsidee** beschreiben;
- die **Umsetzung** der Idee in ein Programm (falls gefordert) erläutern;
- mit genügend **Beispielen** zeigen, dass und wie deine Lösung funktioniert; und
- die wichtigsten Teile des **Quelltextes** anfügen.

Achtung: eine gute Dokumentation muss nicht lang sein – aber ausgedruckt werden (A4, einseitig, jede Aufgabe separat, am besten in je eine Hülle mit Lochrand; keine Mappen!)

Bei Aufgaben mit Programmierung umfasst der **elektronische Teil** den kompletten Quelltext und das ausführbare Programm (Windows, Linux, MacOS oder Android).

Beispiele für Aufgabenbearbeitungen findest du auf den BwInf-Webseiten.

Sende uns die Dokumentationen und eine CD, DVD oder SD-Karte mit den elektronischen Teilen. Beschrifte alles großzügig mit deinem Namen und der Einsendungsnummer, die du bei der Anmeldung erhalten hast. Eine Gruppe schickt ihre Einsendung gemeinsam.

Deine Chancen

Mit einer Teilnahme am Bundeswettbewerb Informatik kannst du nur gewinnen. In allen Runden gibt es **Urkunden** für Teilnahme und besondere Leistungen; zum Dank gibt es kleine **Geschenke** für alle.

Wer sich für die zweite Runde qualifiziert, kann mit Einladungen zu **Informatik-Workshops** rechnen: zum Jugendforum Informatik in Baden-Württemberg, dem Camp „Fit for BwInf“ des Hasso-Plattner-Institus, den Informatiktagen der RWTH Aachen und dem BwInf-Workshop der TU Dortmund. Einige Teilnehmerinnen werden von Google zum **Girls@Google Day** eingeladen.

Nach der zweiten Runde winken die **Forschungstage Informatik** des Max-Planck-Instituts für Informatik. Ausgewählte Gewinner eines zweiten Preises in der zweiten Runde erhalten einen Buchpreis von O'Reilly. Eine Einsendung zur zweiten Runde kann in vielen Bundesländern als **besondere Lernleistung** in die Abiturwertung eingebracht werden.

Die Besten erreichen die **Endrunde**; dort werden Bundessieger und Preisträger ermittelt, die mit **Geldpreisen** belohnt werden. Bundessieger werden in der Regel ohne weiteres Auswahlverfahren in die **Studienstiftung** des deutschen Volkes aufgenommen.