1. Arbeit DiffKurs 8 Informatik - Sporenberg Schuljahr 2011/12 Marl, 27.03.2012

Name: _____ **Klasse:** _____

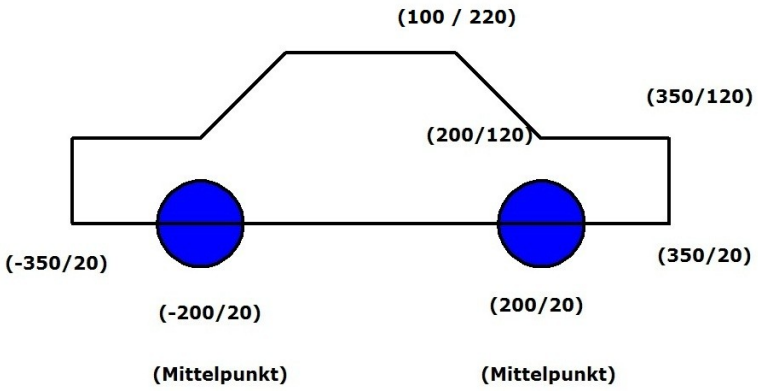
1.Aufgabe: a)Schreibe ein Programm, das ein Quadrat mit der Seitenlänge 100 zeichnet. Dabei soll links gedreht werden. Der Befehl WIEDERHOLE darf hier nicht verwendet werden.

b)Verändere das Programm so, dass damit Quadrate beliebiger Länge gezeichnet werden. Hier muss der Befehl REPEAT verwendet werden.

c)Schreibe ein Programm, das ein gleichseitiges Dreieck mit der Seitenlänge 100 zeichnet. Dabei soll links gedreht werden.

d)Schreibe ein Programm, das ein regelmäßiges Sechseck mit der Seitenlänge 100 zeichnet. Fertige eine Skizze an und überlege dir, wie groß der Winkel sein muss. Dabei soll links gedreht werden.

2.Aufgabe: a) Erläutere den Befehl **setpos** (1. Angabe der Koordinaten durch Zahlen 2. Angabe der Koordinaten durch Variable).
b) Programmiere die nebenstehende Figur. Ergänze die Koordinaten der nicht beschrifteten Punkte auf dem Aufgabenblatt (die linke untere Ecke hat die Koordinaten (-350/20)). Schreibe dazu das entsprechende Programm. Für die Zeichnung muss der Befehl **setpos** verwendet werden.



c) Die linke untere Ecke wird jetzt mit Hilfe zweier Variablen (xko und yko) angesprochen. Schreibe dazu das entsprechende Programm **AutoMitVariable**. Der Aufruf sollte dann wie folgt erfolgen:
AutoMitVariable -350 20

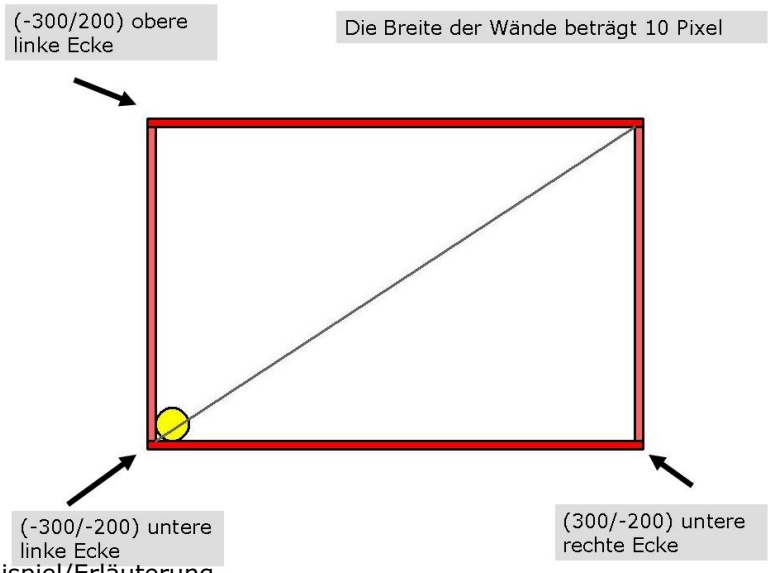
3.Aufgabe: Auf einem waagerechten Holzbrett (s. Abbildung) soll ein Rad von rechts nach links rollen.
a) Schreibe eine Prozedur, die das Brett zeichnet (Breite des Brettes: 800 Pixel, Höhe des Brettes 20 Pixel).

b) Schreibe eine Programm, die das Rad (Radius des Kreises: 40 Pixel) so zeichnet wie in der Abbildung. Dabei soll der Mittelpunkt des Kreises die Koordinaten xko und yko haben. Die Prozedur hat folgende vorgegebene Struktur:
to Kreis :xko :yko
…..end

c) Schreib eine Programm, die den Kreis in der Farbe weiß zeichnet.
d) Schreibe jetzt eine Programm, die den Kreis von rechts nach links bewegt. Überleg dir, welche Koordinate verändert werden muss und begründe dies.

4. Aufgabe: Ein Ball, dargestellt durch einen Kreis, fällt aus einer bestimmten Höhe auf die Wiese und bleibt dort unten liegen.
Unterteile dieses Problem in kleinere Teilprobleme und beschreibe diese ausführlich.
Schreibe dann zu diesen Teilproblemen jeweils die entsprechenden MSWLogo-Prozeduren.

5.Aufgabe: Ein Kreis soll im Rechteck (s. Abbildung) von unten links nach oben rechts bewegt werden.
a)Schreibe ein Programm, das den Rahmen erzeugt (Koordinaten sind angegeben).
b)Schreibe ein Programm, das den Kreis von unten links nach oben rechts bewegt. Der Mittelpunkt des eingezeichneten Kreises ist xko = -270 und yko = -170 mit dem Radius 20 Pixel.



MSWLogo-Befehl	Beispiel/Erläuterung
Allgemein	
;;	Einfügen eines Kommentars
~ (sog. Tilde)	Verkettung zweier Zeilen
print	print "Hallo / erscheint im Commandfeld
cleartext	löscht den Textschirm
readchar, readword, readlist	Dateneingabe (Tastatur, Datei)
keyboardon, keyboardoff	Tastaturbefehle
mouseon, mouseoff, mousepos	Mausbefehle
Turtlegrafik	
setpos	setzt die Turtle auf die angegeb. Position
setheading	setzt die Richtung der Turtle: setheading 90
left, lt, right, rt	Links- bzw. Rechtsdrehung der Turtle
home	setzt die Turtle in Bildmitte ohne zu löschen
forward, fd	fd 100, 100 Pixel vorwärts
back, bk	bk 100, 100 Pixel rückwärts
pos	liefert die aktuelle Position, show pos
xcor, ycor	liefert die x- bzw. y-Koordinate, show xcor
setpencolor, setpc	setpc[255 0 0] - Penfarbe rot
clearscreen, cs	löscht den Bildschirm
setfloodcolor mit fill	füllt ein Polygon mit einer Füllfarbe setfloodcolor[0 255 0]fill - ergibt grün
hideturtle, hat	versteckt die Turtle
showturtle, st	zeigt die Turtle
Prozeduren und Funktionen	
to … end	Beginn … Ende einer Prozedur
make	speichert einen Wert, make "laenge 100
stop	beendet die aktuelle Prozedur
Kontrollstrukturen	
repeat	repeat 5 […..]
if	if :laenge<0 [stop]
ifelse	ifelse :laenge<0 [stop] [repeat 4[…]]
halt	stoppt das Programm
wait	wartet n/60 s, wait 120 -> 120/60 = 2 s

Lösung

1.Aufgabe: a)

```
to quadrat
cs
fd 100 lt 90
fd 100 lt 90
fd 100 lt 90
fd 100 lt 90
end
```

b)

```
to quadrat :laenge
cs
repeat 4[fd :laenge lt 90]
end
```

c)

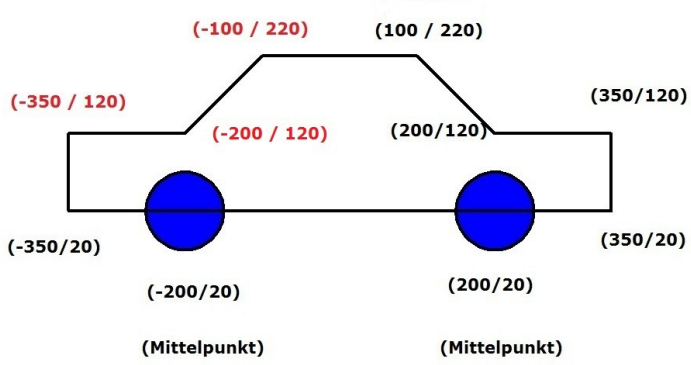
```
to dreieck
cs
repeat 3[fd 100 lt 120]
end
```

d)

```
to sechseck
cs
repeat 6[fd 100 lt 60]
end
```

2.Aufgabe:

a) setpos bei Angabe der Koordinaten über konstante Werte mit eckigen Klammern:
Beispiel -> setpos[-200 100]
Bei Angabe durch Variable mit runden Klammern:
Beispiel -> setpos(list :xko :xko)
Wenn nur eine Koordinate als Variable angegeben wird, so muss ebenfalls mit runden Klammern geschrieben werden:
Beispiel: -> setpos(list 0 :yko)



b)

```
to auto
Cs setpensize[4 4] setpc 0
pu
setpos[-200 20] pd  circle 50 pu
setpos[-200 20] setfc [0 0 255] fill
pu
setpos[200 20] pd  circle 50 pu setpos[200 20] setfc [0 0 255] fill
pu
setpos[-350 20] pd
setpos[350 20] setpos[350 120] setpos[200 120]
setpos[100 220] setpos[-100 220] setpos[-200 120]
```

setpos[-350 120] setpos[-350 20]

ht
end

c) to autoVariable :xko :yko
Cs setpensize[2 2] setpc 0

pu
setpos(list :xko+150 :yko) pd circle 50
pu setpos(list :xko+150 :yko) setfc [0 0 255] fill
pu
setpos(list :xko+550 :yko) pd circle 50
pu setpos(list :xko+550 :yko) setfc [0 0 255] fill

pu
setpos(list :xko :yko) pd setpos(list :xko+700 :yko)
setpos(list :xko+700 :yko+100) setpos(list :xko+550 :yko+100)
setpos(list :xko+450 :yko+200) setpos(list :xko+250 :yko+200)
setpos(list :xko+150 :yko+100) setpos(list :xko :yko+100) setpos(list :xko :yko)

ht
end

3.Aufgabe: a)

to Fahrbahn
setpensize[3 3]
setpc 0
pu
setpos[-400 0] pd
setpos[400 0] setpos[400 20]
setpos[-400 20] setpos[-400 0]
pu setpos[-390 10]
setfc[100 100 220] fill
ht
end

b) Kreis mit schwarzem Rand und gelber Füllung

to kreisBlack :xko :yko
setpc 0 setpensize[3 3]
pu setpos(list :xko :yko)
pd circle 20
pu setpos(list :xko :yko)
setfc[255 255 0]fill
ht
end

c) Kreis mit weißem Rand und weißer Füllung

to kreisWhite :xko :yko
setpc 7 setpensize[3 3]
pu setpos(list :xko :yko)
pd circle 20
pu setpos(list :xko :yko)

setfc 7 fill
ht
end

d)

```
to bewegungHorizontal :xko :yko

if :xko<-250[kreisBlack :xko :yko stop]

kreisBlack :xko :yko
kreisWhite :xko :yko

bewegunghorizontal :xko-1 :yko
end
```

Der besseren Strukturierung und Übersichtlichkeit wegen sollte man ein Hauptprogramm hinzufügen.

```
to mainHorizontal
cs

Fahrbahn
bewegungHorizontal 300 44

end
```

4.Aufgabe:

Teilprobleme sind:

1.Zeichnen des Bodens
2.Zeichnen des Balles (schwarz und weiß)
3.Bewegung nach unten
Die Prozeduren für das Zeichnen der Kreise kann aus der Aufgabe 2 ohne Abstriche übernommen werden.

```
to bewegung :xko :yko

if :yko<40[kreisBlack :xko :yko stop]

kreisBlack :xko :yko
kreisWhite :xko :yko

bewegung :xko :yko-1
;; hier muss die y-Koordinate verringert werden (Bewegung nach unten)
end
```

```
to main
cs
Fahrbahn
bewegung 40 300
end
```

5.Aufgabe: a)

```
to Kiste
cs pu setpensize[3 3]
setpc[0 0 0]

setpos[-300 -200] pd setpos[300 -200]
setpos[300 -190] setpos[-300 -190]
setpos[-300 -200]

pu setpos [-290 -195]
setfc[255 0 0] fill

pu
setpos[-300 200] pd setpos[300 200]
setpos[300 190] setpos[-300 190]
setpos[-300 200]

pu setpos [290 195]
setfc[255 0 0] fill

pu
setpos[-300 -190] pd setpos[-290 -190]
setpos[-290 190] setpos[-300 190]
setpos[-300 -190]

pu setpos [-298 -180]
setfc[255 100 100] fill

pu
setpos[290 -190] pd setpos[300 -190]
setpos[300 190] setpos[290 190]
setpos[290 -190]

pu setpos [295 -180]
setfc[255 100 100] fill

pu setpos[-270 -170]
pd
circle 20
pu setfc[255 255 0] fill

pu setpos[-290 -190]
setpensize[3 3]
setpc[100 100 100]
pd setpos[290 190]

end
```

b)

```
to bewegungHorizontal :xko :yko

if :xko>250[kreisBlack :xko :yko stop]

kreisBlack :xko :yko
```

```
kreisWhite :xko :yko

bewegunghorizontal :xko+1.5 :yko+1
end
```

```
to Kiste
cs pu setpensize[3 3]
setpc[0 0 0]

setpos[-300 -200] pd setpos[300 -200]
setpos[300 -190] setpos[-300 -190]
setpos[-300 -200]


pu setpos [-290 -195]
setfc[255 0 0] fill

pu
setpos[-300 200] pd setpos[300 200]
setpos[300 190] setpos[-300 190]
setpos[-300 200]

pu setpos [290 195]
setfc[255 0 0] fill


pu
setpos[-300 -190] pd setpos[-290 -190]
setpos[-290 190] setpos[-300 190]
setpos[-300 -190]

pu setpos [-298 -180]
setfc[255 100 100] fill



pu
setpos[290 -190] pd setpos[300 -190]
setpos[300 190] setpos[290 190]
setpos[290 -190]

pu setpos [295 -180]
setfc[255 100 100] fill


pu setpos[-270 -170]
pd
circle 20
pu setfc[255 255 0] fill

pu setpos[-290 -190]
setpensize[3 3]
setpc[100 100 100]
pd setpos[290 190]

end
```

```
to kreisBlack :xko :yko
setpc 0 setpensize[3 3]
pu setpos(list :xko :yko)
pd circle 20
pu setpos(list :xko :yko)
setfc[255 255 0]fill
ht
end
```

```
to kreisWhite :xko :yko
setpc 7 setpensize[3 3]
pu setpos(list :xko :yko)
pd circle 20
pu setpos(list :xko :yko)
setfc 7 fill
ht
end
```

```
to main
cs
Kiste
bewegungHorizontal -265 -165
end
```