ARBEITSBLATT ZUR WIEDERHOLUNG DES THEMAS COMPILERBAU

Gegeben sei die Grammatik einer Programmiersprache für Konstruktionen aus der Geometrie mit den folgenden Produktionen:

P = {

PROGRAMM	::= konstruktion name semikolon KONFOLGE ende semikolon
KONFOLGE	::= KONSTRUKTION | KONSTRUKTION semikolon KONFOLGE
KONSTRUKTION	::= KREIS | STRECKE | PUNKT
KREIS	::= kreis klammerauf PUNKT semikolon AUSDRUCK klammerzu
STRECKE	::= strecke klammerauf PUNKT semikolon PUNKT klammerzu
PUNKT	::= punkt klammerauf AUSDRUCK semikolon AUSDRUCK klammerzu
AUSDRUCK	::= TERM | TERM strichoperator AUSDRUCK
TERM	::= FAKTOR | FAKTOR punktoperator TERM
FAKTOR	::= name | zahl | klammerauf AUSDRUCK klammerzu

}

a) *Geben Sie die Menge der **Terminalen** und **Nichtterminalen** an.*

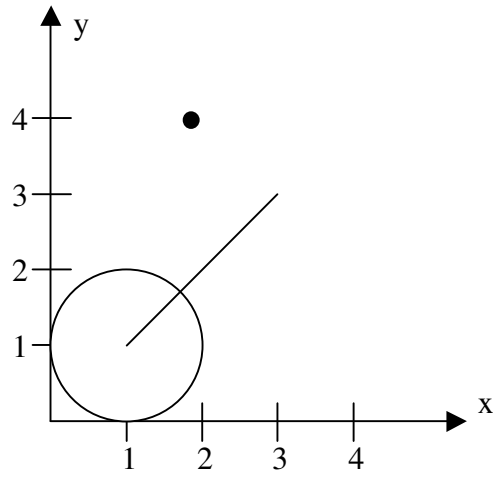
b) Gegeben ist das folgende Programm dieser Programmiersprache:

```
construction Figur;
  circle(point(2;5);2);
  line(point(2;3);point(2;3-2));
  point(2;3)
end;
```

Zeichnen Sie das Bild, welches die Programmiersprache im ersten Quadranten ausgeben würde.

c) Rechts abgebildete Zeichnung wurde erstellt.

Implementieren Sie das zugehörige Programm in der gegebenen Programmiersprache.



d) Im folgenden Programm sind Fehler enthalten:

```
construktion Fehler#
  circle(point(a;b);c)
  line((2;3)-(2;2));
  point(2;1;13);
  circle(point(1;1);-10)
end.
```

Im Unterricht haben wir im Zusammenhang mit dem Compiler über drei verschiedene Arten von Fehlern gesprochen: (1) Laufzeitfehler,
(2) Syntaxfehler und (3) lexikalischer Fehler.

Erläutern Sie jeweils, was man unter diesen Fehler versteht.

Geben Sie alle Fehler des oben abgebildeten Programms an und analysieren Sie die Art der jeweiligen Fehler.

Geben Sie das Programm korrekt an.

ARBEITSBLATT ZUR WIEDERHOLUNG DES THEMAS COMPILERBAU

e) Die Programmiersprache soll um die Möglichkeit einer ZUWEISUNG erweitert werden. Damit wären folgende Befehle denkbar:

```
construction Test;
  a := 5;
  b := point(a;9);
  c := line(point(2;1);point(a;11));
  d := circle(point(1;1);a);
  draw(b);
  draw(c);
  draw(d)
end;
```

Geben Sie alle Änderungen in der Grammatik an.

f) Im Unterricht haben wir den Prozess des Parsens mithilfe einer Tabelle durchgeführt:

Produktionsregel	**Zeichenkette**	**Aktion**	**Token**
		erstes Token holen:	konstruktion
KONSTRUKTION	`construction`	ist ok, also hole nächstes Token:	name
	`Test`	ist ok, also hole nächstes Token:	...
	...	...	...

Führen Sie diese Tabelle für das folgende Programm zu Ende:

```
construction Test;
  circle(point(2;5);2*3)
end;
```

g) *Zeichnen Sie den Ableitungsbaum für das Konstruktionsprogramm aus Teilaufgabe f.*

h) Im Unterricht haben wir für unsere Minisprache einen endlichen Automaten entwickelt. Dieser konnte den Quelltext mithilfe einer vordefinierten Symboltabelle in die einzelnen Tokens zerlegen und in einer Tokenliste speichern.

Entwickeln Sie eine mögliche Symboltabelle und den zugehörigen Scannerautomat, der für diese Konstruktions-Programmiersprache entwickelt werden müsste.

Geben Sie anschließend die Symboltabelle und die Tokenliste für das Konstruktionsprogramm aus Teilaufgabe f an.

i) So und nun implementieren Sie einen Compiler für diese Konstruktions-Programmiersprache ... ;-)