

Brandenburgische Technische Universität Cottbus
Institut für Informatik
LS „Datenstrukturen und Softwarezuverlässigkeit“

Dokumentation PED Version 4.3

Dateiformat

Reimo Tiedemann

4. Juni 1997

Reimo Tiedemann
Dokumentation PED Version 4.3
Dateiformat

Brandenburgische Technische Universität Cottbus
Institut für Informatik
LS „Datenstrukturen und Softwarezuverlässigkeit“
Postfach 10 13 44
D - 03013 Cottbus
Tel: (+49-355) 69 3884
Fax: (+49-355) 69 3820
Email: ped@informatik.tu-cottbus.de
WWW: <http://www-dssz.informatik.tu-cottbus.de/~wwwdssz>

Inhaltsverzeichnis

1	Dateiformat	5
2	Ein Beispielnetz	13

1 Dateiformat

Als Dateiformat zur externen Speicherung von Petrinetzen wurde ein ASCII-Format gewählt, so daß eine PED-Datei (Endung .ped) prinzipiell von jedem Nutzer lesbar und eine Anbindung an weitere Petrinetztools möglich ist. Zur Identifizierung einer PED-Datei muß in der ersten Zeile der Datei “# PED Vx.x petri net” angegeben werden. Da das Dateiformat zwischenzeitlich mehrfach erweitert wurde, ist die Angabe der Versionsnummer (im aktuellen Fall “V4.3”) für ein korrektes Einlesen anzugeben. Damit ist auch für ältere PED-Versionen Abwärtskompatibilität gewährleistet.

Innerhalb der Datei können beliebig Kommentare eingefügt werden. Diese beginnen immer mit einem Doppelkreuz (#) und erstrecken sich bis zum Zeilenende. Damit können Generatoren oder Filterprogramme ihre Arbeit dokumentieren. Als Trennzeichen werden Leerzeichen oder Zeilenumbrüche verwendet. Mehrfach hintereinander auftretende Leerzeichen bzw. Zeilenumbrüche werden als ein einziges Trennzeichen interpretiert (eine Ausnahme von all dem sind die "String"-Umgebungen, siehe unten).

Kommentare, Coarse-Knoten, Plätze und Transitionen eines Teilnetzes werden innerhalb eines Bereiches gruppiert. Sie sind *nicht* hierarchisch angeordnet. Die Netzhierarchie läßt sich über die vergebenen Netznummern und die entsprechenden *SubNetNumber*-Attribute der Coarse-Knoten rekonstruieren. Alle Bögen sind in einem einheitlichen Bereich am Ende der Datei separat aufgeführt. Ihre *Pre*- und *PostNodes* werden durch *Element-Identifier* referenziert (siehe unten).

Die Beschreibung des Dateiformats liegt in einer BNF vor. Diese ist soweit wie möglich informell gehalten. Nichtterminalsymbole sind in spitzen Klammern ($\langle \text{Nichtterminalsymbol} \rangle$) angegeben. Terminalsymbole werden **fett** gedruckt, “_” steht für ein Leerzeichen und “ \backslash ” kennzeichnet einen Zeilenumbruch. Beliebig oft auftretende (optionale) Sequenzen sind in geschweifte Klammern gesetzt. Bei einer Auswahl zwischen mehreren Ausdrücken werden diese durch “|” getrennt.

```
 $\langle \text{File} \rangle$            :=  $\langle \text{File\_Description} \rangle$   
                     $\langle \text{Options} \rangle$   
                     $\langle \text{TopLevel\_Page} \rangle$   
                    {  $\langle \text{Page} \rangle$  }  
                    {  $\langle \text{Arc} \rangle$  }
```

```
 $\langle \text{File\_Description} \rangle$  := #_PED_V4.3_petri_net  $\backslash$ 
```

Im *Options*-Block werden die aktuellen Einstellungen des Editors gespeichert. Den Bezeichnern der einzelnen Parameter läßt sich im allgemeinen auch ihre Bedeutung entnehmen. Bei **Size*, **Length* und **Space*-Parametern werden Pixelwerte angegeben. *HierFlag*, *NetItemLength*,

NetItemSpace und *LevelSpace* beschreiben Einstellungen innerhalb des Hierarchiebrowsers (horizontale bzw. vertikale Darstellung, Größe und Abstände zwischen den Netz-Icons). Die *Default**-Parameter geben die einstellbaren Defaultwerte für die einzelnen Attribute an, die *Show**-Parameter ihre momentane Sichtbarkeit innerhalb des Editors. Die **Pages*-Parameter speichern die Größe der eingestellten Arbeitsfläche (Anzahl der horizontalen bzw. vertikalen Seiten).

<Options> := **GridSize**=*<Unsigned_Int>* ^[n]
NodeSize=*<Unsigned_Int>* ^[n]
GridFlag=*<Bool_Flag>* ^[n]
SnapFlag=*<Bool_Flag>* ^[n]
HierFlag=*<Bool_Flag>* ^[n]
NetItemLength=*<Unsigned_Int>* ^[n]
NetItemSpace=*<Unsigned_Int>* ^[n]
LevelSpace=*<Unsigned_Int>* ^[n]
DefaultCapacity=*<Capacity>* ^[n]
DefaultTokens=*<Mark>* ^[n]
DefaultPriority=*<Priority>* ^[n]
DefaultMultiplicity=*<Multiplicity>* ^[n]
DefaultPlaceTime=*<Delay>* ^[n]
DefaultTransitionTime=*<Delay>* ^[n]
DefaultArcTime=*<Delay>* ^[n]
DefaultPlaceInterval=*<Interval>* ^[n]
DefaultTransitionInterval=*<Interval>* ^[n]
DefaultArcInterval=*<Interval>* ^[n]
ShowNumberFlag=*<Bool_Flag>* ^[n]
ShowNameFlag=*<Bool_Flag>* ^[n]
ShowAlgebraFlag=*<Bool_Flag>* ^[n]
ShowCapacityFlag=*<Bool_Flag>* ^[n]
ShowTokensFlag=*<Bool_Flag>* ^[n]
ShowPriorityFlag=*<Bool_Flag>* ^[n]
ShowMultiplicityFlag=*<Bool_Flag>* ^[n]
ShowPlaceTimeFlag=*<Bool_Flag>* ^[n]
ShowTransitionTimeFlag=*<Bool_Flag>* ^[n]
ShowArcTimeFlag=*<Bool_Flag>* ^[n]
ShowPlaceIntervalFlag=*<Bool_Flag>* ^[n]
ShowTransitionIntervalFlag=*<Bool_Flag>* ^[n]
ShowArcIntervalFlag=*<Bool_Flag>* ^[n]
ShowCommentFlag=*<Bool_Flag>* ^[n]
ShowBasicObjectsFlag=*<Bool_Flag>* ^[n]
ShowConflictClusterFlag=*<Bool_Flag>* ^[n]
HorizontalPages=*<Unsigned_Int>* ^[n]
VerticalPages=*<Unsigned_Int>* ^[n]

<TopLevel_Page> := *<Element>* { *<Element>* }

<Page> := { *<Element>* }

$\langle Element \rangle := \langle Comment \rangle \mid \langle Coarse Node \rangle \mid \langle Place \rangle \mid \langle Transition \rangle$

$\langle Comment \rangle := \mathbf{C}_{\lfloor} \{ \lfloor \langle NetID \rangle \lfloor \langle XY_Pos \rangle \lfloor \langle Text \rangle \lfloor \}$

Netzelemente können neben ihren strukturellen Eigenschaften mehrere graphische Ausprägungen besitzen. Dies ist dann der Fall, wenn ein Knoten als Kopplungsknoten in anderen Teilnetzen dargestellt wird. In den nun folgenden Definitionen der einzelnen Netzelemente sind die jeweiligen graphischen Ausprägungen in einem *Graphics-Block zusammengefaßt.

Logische Knoten sind als separate Elemente aufgeführt, sie werden also *nicht* als graphische Ausprägungen eines Elementes behandelt. Die strukturellen Eigenschaften dieser Knoten sind zwar gleich, sie werden aber trotzdem mehrfach angegeben. Logische Beziehungen zwischen den Knoten lassen sich nur über ihre Namensgleichheit (bzw. auch Nummerngleichheit) wiederherstellen.

$\langle Coarse Node \rangle := \mathbf{V}_{\lfloor} \{ \lfloor \langle Number \rangle \lfloor \langle Name \rangle \lfloor \langle SubNetNumber \rangle \lfloor \langle CN_Type \rangle \lfloor \langle CN_Graphics \rangle \lfloor \}$

$\langle CN_Graphics \rangle := \langle Mode_0 \rangle \lfloor \langle ElemID \rangle \lfloor \langle NetID \rangle \lfloor \langle XY_Pos \rangle \lfloor \langle XY_Attr_Offs \rangle \lfloor$

$\langle Place \rangle := \mathbf{P}_{\lfloor} \{ \lfloor \langle Number \rangle \lfloor \langle Name \rangle \lfloor \langle Capacity \rangle \lfloor \langle Mark \rangle \lfloor \langle Delay \rangle \lfloor \langle Interval \rangle \lfloor \langle P_Type \rangle \lfloor \langle Algebra \rangle \lfloor \langle P_Graphics \rangle \lfloor \}$

$\langle P_Graphics \rangle := \langle Mode_0_1 \rangle \lfloor \langle ElemID \rangle \lfloor \langle NetID \rangle \lfloor \langle XY_Pos \rangle \lfloor \langle XY_Attr_Offs \rangle \lfloor \langle XY_Alg_Offs \rangle \lfloor \{ \langle Mode_2 \rangle \lfloor \langle ElemID \rangle \lfloor \langle NetID \rangle \lfloor \langle XY_Pos \rangle \lfloor \langle XY_Attr_Offs \rangle \lfloor \langle XY_Alg_Offs \rangle \lfloor \}$

$\langle Transition \rangle := \mathbf{T}_{\lfloor} \{ \lfloor \langle Number \rangle \lfloor \langle Name \rangle \lfloor \langle Priority \rangle \lfloor \langle Delay \rangle \lfloor \langle Interval \rangle \lfloor \langle T_Type \rangle \lfloor \langle T_Shape \rangle \lfloor \langle Algebra \rangle \lfloor \langle T_Graphics \rangle \lfloor \}$

$\langle T_Graphics \rangle := \langle Mode_0_1 \rangle \lfloor \langle ElemID \rangle \lfloor \langle NetID \rangle \lfloor \langle XY_Pos \rangle \lfloor \langle XY_Attr_Offs \rangle \lfloor \langle XY_Alg_Offs \rangle \lfloor \langle T_Pos \rangle \lfloor \{ \langle Mode_2 \rangle \lfloor \langle ElemID \rangle \lfloor \langle NetID \rangle \lfloor \langle XY_Pos \rangle \lfloor \langle XY_Attr_Offs \rangle \lfloor \langle XY_Alg_Offs \rangle \lfloor \langle T_Pos \rangle \lfloor \}$

$$\langle \text{Arc} \rangle \quad := \mathbf{A}_{\square} \{ \square \langle \text{Multiplicity} \rangle \square \langle \text{Delay} \rangle \square \langle \text{Interval} \rangle \square \langle \text{A_Type} \rangle \square \langle \text{A_Shape} \rangle \square \langle \text{Algebra} \rangle \square \langle \text{A_Graphics} \rangle \}$$

$$\langle \text{A_Graphics} \rangle \quad := \langle \text{Mode}_{0-1} \rangle \square \langle \text{ElemID} \rangle \square \langle \text{NetID} \rangle \square \langle \text{PreNode} \rangle \square \langle \text{PostNode} \rangle \square \langle \text{XY_Attr_Offs} \rangle \square \langle \text{XY_Alg_Offs} \rangle \square \langle \text{Knots} \rangle \square \{ \langle \text{Mode}_{2-3} \rangle \square \langle \text{ElemID} \rangle \square \langle \text{NetID} \rangle \square \langle \text{PreNode} \rangle \square \langle \text{PostNode} \rangle \square \langle \text{XY_Attr_Offs} \rangle \square \langle \text{XY_Alg_Offs} \rangle \square \langle \text{Knots} \rangle \square \}$$

$$\langle \text{PreNode} \rangle \quad := \langle \text{N_Type} \rangle \square \langle \text{Number} \rangle \square \langle \text{ElemID} \rangle \square \langle \text{NetID} \rangle$$

$$\langle \text{PostNode} \rangle \quad := \langle \text{N_Type} \rangle \square \langle \text{Number} \rangle \square \langle \text{ElemID} \rangle \square \langle \text{NetID} \rangle$$

$$\langle \text{Knots} \rangle \quad := \langle \# \text{Knots} \rangle \square \langle \text{XY_Pos} \rangle \square \langle \text{XY_Pos} \rangle \{ \square \langle \text{XY_Pos} \rangle \} \square$$

#Knots gibt die Anzahl der folgenden Punkte eines Kantenzuges an. Sie beträgt mindestens 2, wobei der erste Punkt der Berührungspunkt mit dem *PreNode* und der letzte der mit dem *PostNode* ist.

$$\langle \# \text{Knots} \rangle \quad := \langle \text{Unsigned_Int} \rangle$$

XY_Pos beschreibt die Koordinaten eines Punktes auf der Zeichenfläche des Editors in Pixel. Der erste Wert ist die X-, der zweite die Y-Koordinate. Der Ursprung des Koordinatensystems liegt links oben. Die Position eines Knotens ist als dessen Mittelpunkt angegeben.

$$\langle \text{XY_Pos} \rangle \quad := \langle \text{Unsigned_Int} \rangle \square \langle \text{Unsigned_Int} \rangle$$

XY_Attr_Offs gibt die Position des Attributblocks als *Offset* zum Mittelpunkt eines Netzelementes an. Bei Bögen wird als "Mittelpunkt" die genaue Mitte zwischen dem *PreNode* und dem *PostNode* verwendet.

$$(x_m, y_m) = ((x_{pre} + x_{post})/2, (y_{pre} + y_{post})/2)$$

$$\langle \text{XY_Attr_Offs} \rangle \quad := \langle \text{Int} \rangle \square \langle \text{Int} \rangle$$

Entsprechend handelt es sich bei *XY_Alg_Offs* um die Position des *Algebra*-Attributs als *Offset* zum Mittelpunkt eines Netzelementes (siehe oben).

$$\langle \text{XY_Alg_Offs} \rangle \quad := \langle \text{Int} \rangle \square \langle \text{Int} \rangle$$

$$\langle \text{Name} \rangle \quad := \text{"einzeiliger String"}$$

$$\langle \text{Text} \rangle \quad := \text{"mehrzeiliger String"}$$

$$\langle \text{Algebra} \rangle \quad := \text{"mehrzeiliger String"}$$

Jeder Knoten besitzt eine Nummer. Sie ist innerhalb eines Knotentyps eindeutig, sofern es keine logischen Knoten gibt (deren Nummern sind gleich).

$\langle \text{Number} \rangle \quad := \langle \text{Unsigned_Int} \rangle$

SubNetNumber gibt die *NetID* des darunterliegenden Teilnetzes für einen *Coarse Node* an.

$\langle \text{SubNetNumber} \rangle \quad := \langle \text{Unsigned_Int} \rangle$

$\langle \text{Capacity} \rangle \quad := \langle \text{Unsigned_Int} \rangle$

$\langle \text{Mark} \rangle \quad := \langle \text{Unsigned_Int} \rangle$

$\langle \text{Delay} \rangle \quad := \langle \text{Unsigned_Int} \rangle$

$\langle \text{Interval} \rangle \quad := [\langle \text{Unsigned_Int} \rangle , \langle \text{Unsigned_Int} \rangle]$

$\langle \text{Priority} \rangle \quad := \langle \text{Unsigned_Int} \rangle$

$\langle \text{Multiplicity} \rangle \quad := \langle \text{Unsigned_Int} \rangle$

Jedes Netzelement besitzt einen eindeutigen *Identifier*.

$\langle \text{ElemID} \rangle \quad := \mathbf{I_} \langle \text{Unsigned_Int} \rangle$

Jedes Teilnetz besitzt einen eindeutigen *Identifier*. Das Toplevel-Netz hat immer die ID 0.

$\langle \text{NetID} \rangle \quad := \mathbf{N_} \langle \text{Unsigned_Int} \rangle$

Mode_0 impliziert, daß es keine weiteren graphischen Ausprägungen des Netzelementes geben kann.

$\langle \text{Mode}_0 \rangle \quad := \mathbf{M_} \mathbf{0}$

Bei *Mode_0_1* handelt es sich um das eigentliche strukturelle Netzelement (Originalelement). Der Parameter gibt an, ob es noch weitere graphische Ausprägungen gibt (Kopplungselemente). Bei Bögen im *Mode 1* liegen *PreNode* und *PostNode* in verschiedenen Teilnetzen. Solch ein Bogen wird im Editor nicht gezeichnet.

Parameter : 0 ... keine weiteren graph. Ausprägungen
 1 ... weitere graph. Ausprägungen vorhanden

$\langle \text{Mode}_0_1 \rangle \quad := \mathbf{M_} (\mathbf{0} \mid \mathbf{1})$

Mit *Mode_2* wird eine weitere graphische Ausprägung des Originalknotens in Form eines Kopplungsknotens gekennzeichnet.

$\langle Mode_2 \rangle := \mathbf{M_2}$

Mode_2_3 wird nur für Bögen verwendet.

Parameter : 2 ... Bogen zwischen einem Originalknoten (oder einem Vertreter-Coarse-Knoten) und einem Kopplungsknoten
 3 ... Bogen zwischen einem Originalknoten (oder dessen obersten Vertreter-Coarse-Knoten) und einem Coarse-Knoten

$\langle Mode_2_3 \rangle := \mathbf{M_ (2 | 3)}$

CN_Type spezifiziert den Typ des Coarse-Knotens, bzw. dessen Darstellung im Editor genauer.

Parameter : V ... gemischtberandetes Teilnetz (bzw. Standardversion)
 P ... platzberandetes Teilnetz
 T ... transitionsberandetes Teilnetz

$\langle CN_Type \rangle := \mathbf{V | P | T}$

N_Type gibt allgemein einen Knotentyp an.

Parameter : V ... *Coarse Node*
 P ... Platz
 T ... Transition

$\langle N_Type \rangle := \mathbf{V | P | T}$

P_Type wird nur für Plätze verwendet und dient als *flag* zur Kennzeichnung von logischen Plätzen.

Parameter : 0 ... normal
 1 ... logisch

$\langle P_Type \rangle := \mathbf{0 | 1}$

T_Type wird nur für Transitionen verwendet und dient zur Kennzeichnung von Transitionstypen (logisch, Fakt) kombiniert mit zulässigen Darstellungsformen (normal, gefüllt).

Parameter : 0 ... normal
 1 ... logisch
 2 ... Fakt
 3 ... Fakt und logisch
 4 ... gefüllt
 5 ... gefüllt und logisch

$\langle T_Type \rangle := \mathbf{0 | 1 | 2 | 3 | 4 | 5}$

T_Shape wird nur für Transitionen verwendet und beschreibt die äußere Form einer Transition.

Parameter : 0 ... quadratisch
 1 ... rechteckig
 2 ... linienförmig

$\langle T_Shape \rangle := \mathbf{0} \mid \mathbf{1} \mid \mathbf{2}$

T_Pos wird nur für Transitionen verwendet und beschreibt die Lage einer Transition.

Parameter : 0 ... waagrecht
 1 ... 0 um 90° gedreht
 2 ... 0 um 135° gedreht
 3 ... 0 um 45° gedreht

$\langle T_Pos \rangle := \mathbf{0} \mid \mathbf{1} \mid \mathbf{2} \mid \mathbf{3}$

A_Type wird nur für Bögen verwendet und beschreibt den Bogentyp (Darstellung der Kanten-
 spitze).

Parameter : 0 ... Standardkante (einfacher Pfeil)
 1 ... Inhibitor-kante (Kreisbogen)
 2 ... Abräumkante (doppelter Pfeil)
 3 ... Testkante (gefüllter Kreisbogen)

$\langle A_Type \rangle := \mathbf{0} \mid \mathbf{1} \mid \mathbf{2} \mid \mathbf{3}$

A_Shape wird nur für Bögen verwendet und beschreibt die Darstellungform des Kantenzuges.

Parameter : 0 ... durchgezogen
 1 ... gestrichelt
 2 ... gepunktet
 3 ... punktgestrichelt
 4 ... durchgezogen und teilpunktiert

$\langle A_Shape \rangle := \mathbf{0} \mid \mathbf{1} \mid \mathbf{2} \mid \mathbf{3} \mid \mathbf{4}$

$\langle Bool_Flag \rangle := \mathbf{0} \mid \mathbf{1}$

2 Ein Beispielnetz

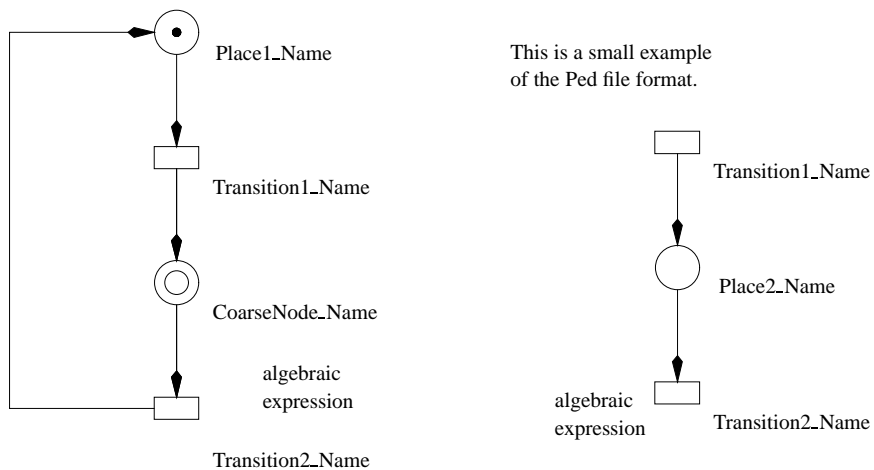


Abbildung 2.1. „TopLevel“ und Subnetz: „CoarseNode_Name“

```
# PED V4.3 petri net

GridSize=15
NodeSize=8
GridFlag=1
SnapFlag=1
HierFlag=1
NetItemLength=40
NetItemSpace=15
LevelSpace=20
DefaultCapacity=0
DefaultTokens=0
DefaultPriority=0
DefaultMultiplicity=1
DefaultPlaceTime=0
DefaultTransitionTime=0
DefaultArcTime=0
DefaultPlaceInterval=[0,0]
DefaultTransitionInterval=[0,0]
DefaultArcInterval=[0,0]
ShowNumberFlag=0
ShowNameFlag=1
ShowAlgebraFlag=1
```

```

ShowCapacityFlag=0
ShowTokensFlag=1
ShowPriorityFlag=0
ShowMultiplicityFlag=1
ShowPlaceTimeFlag=0
ShowTransitionTimeFlag=0
ShowArcTimeFlag=0
ShowPlaceIntervalFlag=0
ShowTransitionIntervalFlag=0
ShowArcIntervalFlag=0
ShowCommentFlag=1
ShowBasicObjectsFlag=0
ShowConflictClusterFlag=0
HorizontalPages=2
VerticalPages=2

# Elements of page 0 (TopLevel)

V { 0
  "CoarseNode_Name" 1 P
  M 0 I 10 N 0 135 150 13 13
}

P { 0
  "Place1_Name" 0 1 0 [0,0] 0
  ""
  M 0 I 1 N 0 135 60 14 10 35 -40
}

T { 0
  "Transition1_Name" 0 0 [0,0] 0 1
  ""
  M 1 I 3 N 0 135 105 13 13 35 -40 0
  M 2 I 15 N 1 135 105 13 13 35 -40 0
}

T { 1
  "Transition2_Name" 0 0 [0,0] 0 1
  "algebraic
expression"
  M 1 I 4 N 0 135 195 13 21 31 -10 0
  M 2 I 13 N 1 135 195 13 13 -83 -3 0
}

# End page 0 (TopLevel)

# Elements of page 1 (CoarseNode_Name)

C {
  N 1 75 75
  "This is a small example
of the Ped file format."
}

```

```

P { 1
  "Place2_Name" 0 0 0 [0,0] 0
  ""
  M 0 I 2 N 1 135 150 15 9 -77 0
}

# End page 1 (CoarseNode_Name)

# ***** Arcs *****

A {
  1 0 [0,0] 0 0
  ""
  M 0 I 5 N 0 P 0 I 1 N 0 T 0 I 3 N 0 10 10 0 40 2 135 68 135 101
}

A {
  1 0 [0,0] 0 0
  ""
  M 1 I 6 N 0 T 0 I 3 N 0 P 1 I 2 N 1 10 10 0 40 2 135 109 135 142
  M 2 I 16 N 1 T 0 I 15 N 1 P 1 I 2 N 1 10 10 0 40 2 135 109 135 142
  M 3 I 12 N 0 T 0 I 3 N 0 V 0 I 10 N 0 10 10 0 40 2 135 109 135 142
}

A {
  1 0 [0,0] 0 0
  ""
  M 1 I 7 N 0 P 1 I 2 N 1 T 1 I 4 N 0 10 10 0 40 2 135 158 135 191
  M 2 I 14 N 1 P 1 I 2 N 1 T 1 I 13 N 1 10 10 0 40 2 135 158 135 191
  M 3 I 11 N 0 V 0 I 10 N 0 T 1 I 4 N 0 10 10 0 40 2 135 158 135 191
}

A {
  1 0 [0,0] 0 0
  ""
  M 0 I 9 N 0 T 1 I 4 N 0 P 0 I 1 N 0 10 10 0 40 4 \
127 195 75 195 75 60 127 60
}

```