III プログラム言語の文法(構文図)　～続き～

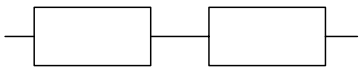
４．構文図(Syntax Chart)

BNF は構文の定義に適しているが、必ずしも読みやすい（視認性が高い）とは言えない。構文図は、BNF に比べ、より視覚的にプログラム言語の文法を記述する手法である。

４．１ BNF との関係

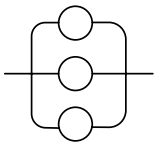
(1)シーケンス

２桁の数字 ::= 数字　数字



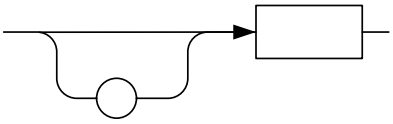
(2)選択

変数 ::= ‘x’ | ‘y’ | ‘z’



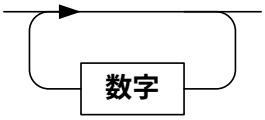
(3)省略

１桁整数 := [‘－’]　数字



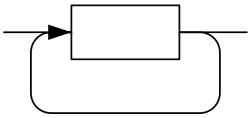
(4)ZERO 回以上の繰り返し

数字列 ::= 数字＊



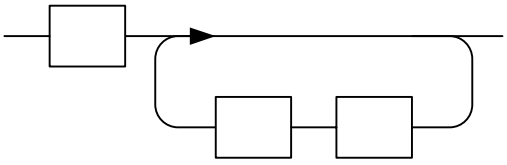
(5)１回以上の繰り返し

数字列 ::= 数字＋



注）繰り返しは矢印の根本から先へ順に実行される

文 ::= 文１　＜ 文２　文３ ＞＊



注) ノードは構文図の最小単位でありトークンとも呼ばれる

練習４

次に示す BNF 記述を、構文図を用いて表せ。ここで英文字と数字および式は既に定義されているとする。

(1) 数字のみで表記されたパスワード ::= 数字＊

(2) 英文字で始まり、その後に数字が続くパスワード ::= 英文字　数字＋

数字

(3) 英文字と数字からなるパスワード ::= ＜ 英文字 ｜ 数字 ＞＋

(4) 英文字から始まり、英文字と数字からなるパスワード

::= 英文字　＜英文字 ｜ 数字 ＞＊

(5) 英数字３桁のパスワード

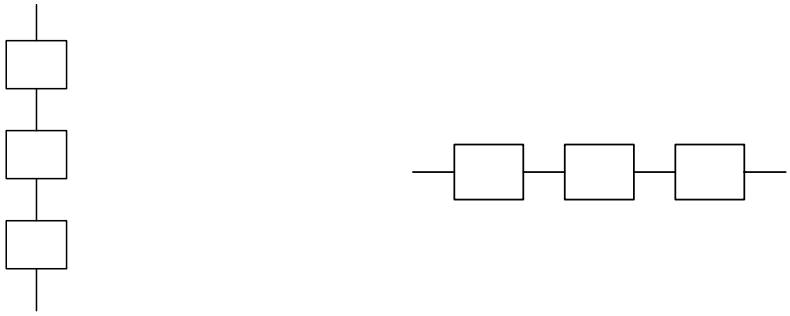
::= ＜英文字 ｜ 数字 ＞ ｜ ＜英文字 ｜ 数字 ＞ ｜ ＜英文字 ｜ 数字 ＞

４．２ プログラム構造との関係

文法と構造では考え方が違うので注意すること。ここではプログラムの流れを構文図で示すことができることを示している。

(1)直接

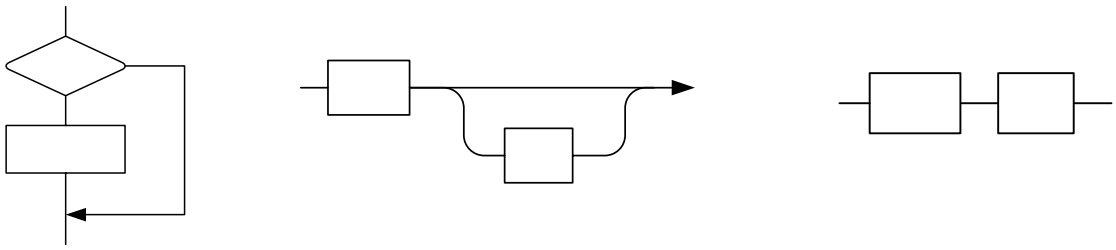
直接 ::= 文１　文２　文３



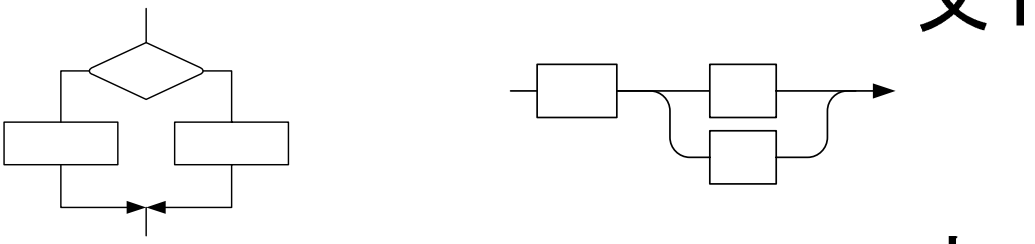
(2)分岐

if 文 ::= 条件　[文]

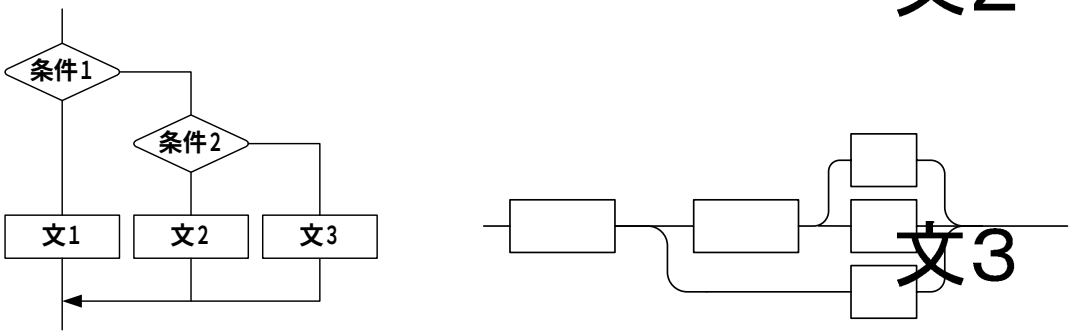
(注：これはあくまで例である。実際は if 構文として「文」は省略できない。つまり構文図はプログラムの流れではなく、あくまでプログラムの構造を示している)



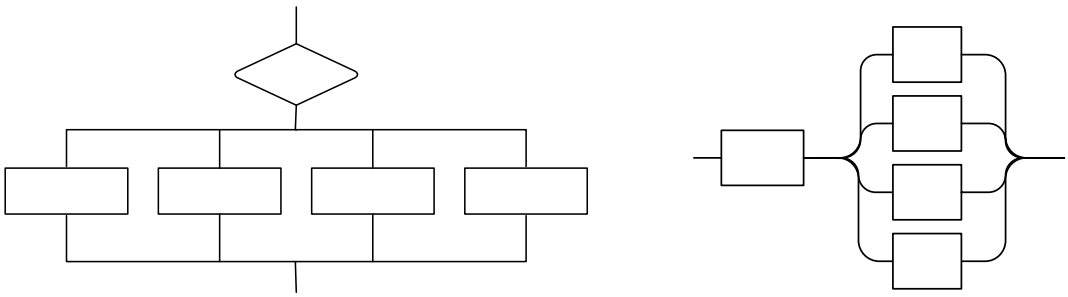
if 文 ::= 条件　文１　｜　文２



if 文 ::= 条件１　文１　＜ 条件２　文２　｜　文３ ＞



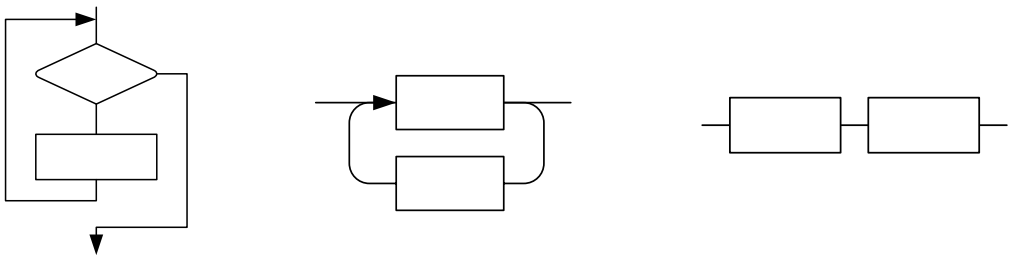
if 文 ::= 条件　＜ 文１　｜　文２　｜　文３　｜　文４ ＞



(3)繰り返し

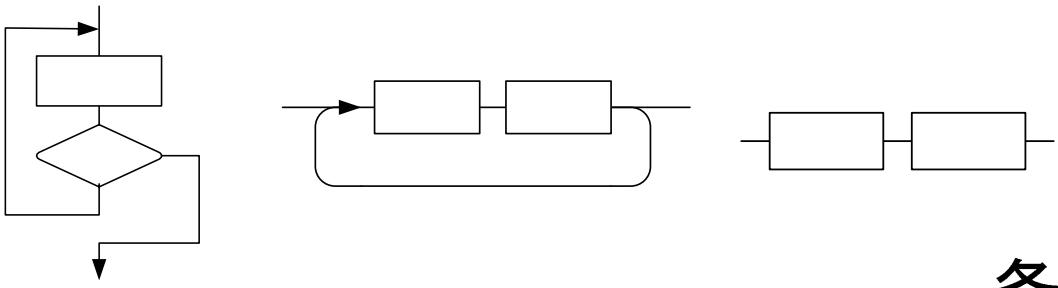
前判定

繰り返し ::= 条件　＜ 文　条件 ＞＊



後判定

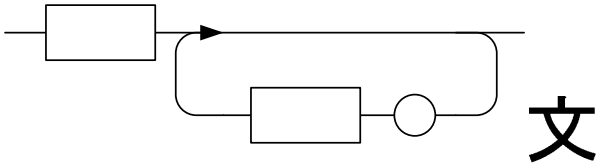
繰り返し ::= ＜ 文　条件 ＞＋



繰り返し時の順番

繰り返し時の順番は矢印の根本から先に順に記述する

繰り返し ::= 変数　＜ ‘,’　変数 ＞＊



４．３ Ｃ言語構文への適応例

(1)if 文

Ｃ言語の簡単な if 文は

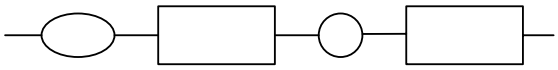
if(条件)

文

BNF では

if 文 ::= ‘if(‘　条件　‘)’　文

構文図では



(2)for 文

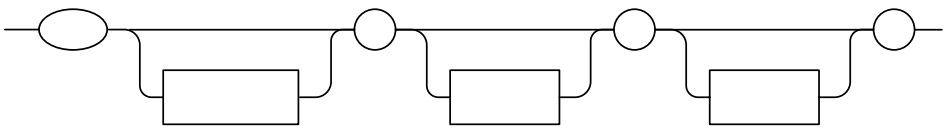
Ｃ言語の for 文は

for(初期条件 ; 条件 ; 更新)

BNF では

for 文 ::= ‘for(‘　[初期条件]　‘;’　[条件]　‘;’　[更新]　‘)’

構文図では



(3)整数

Ｃ言語で整数は BNF では

正の整数 ::= [‘＋’]　ZERO 無し数字　数字＊

構文図では