Example 1: ABNF grammar for U.S. postal addresses (taken from Augmented Backus–Naur form - Wikipedia)

```
; U.S. Postal Addresses Grammar
; This grammar defines the syntax of a U.S. postal address.

postal-address   = name-part street zip-part

; Name part of the address
name-part        = *(personal-part SP) last-name [SP suffix] CRLF
name-part CRLF = *(personal-part SP) last-name [SP suffix] ;
Incorrect: Element misplacement (‘CRLF’ should not be next to
‘name-part’).
name-part        =/ personal-part CRLF

personal-part    = first-name / (initial ".")
first-name       = *ALPHA
first-name = *ALPHA "." 1*DIGIT ; Incorrect: Invalid digit characters
in first name.
initial          = ALPHA
last-name        = *ALPHA
suffix           = ("Jr." / "Sr." / 1*("I" / "V" / "X"))

; Street part of the address
street           = [apt SP] house-num SP street-name CRLF
apt              = 1*4DIGIT
house-num        = 1*8(DIGIT / ALPHA)
street-name      = 1*VCHAR

; ZIP part of the address
zip-part         = town-name "," SP state 1*2SP zip-code CRLF
town-name        = 1*(ALPHA / SP)
state            = 2ALPHA
zip-code         = 5DIGIT ["-" 4DIGIT]
```

Example 2: ABNF grammar for simple arithmetic expressions (addition, subtraction, multiplication, division)

```
; Simple Arithmetic Expressions Grammar
; This grammar defines the syntax of a simple arithmetic expression
with addition, subtraction, multiplication, and division.

expression = term *(addition term / subtraction term)
expression = term (addition term / subtraction term) ; Incorrect: This
only allows for one occurrence of either addition term or subtraction
term.
term = factor *(multiplication factor / division factor)
factor = number / "(" expression ")"
addition = "+"
subtraction = "-"
multiplication = "*"
division = "/"
number = 1*DIGIT

; Terminal definitions
DIGIT = %x30-39 ; DIGIT represents any character in the range %x30-39,
which corresponds to the ASCII digits 0 through 9.
DIGIT = 0-9 ; Incorrect: Character ranges must be represented using
the %x notation followed by hexadecimal values for the characters.
```

Example 3: ABNF grammar for IPv4 addresses

```
; IPv4 Address Grammar
; This grammar defines the syntax of an IPv4 address.

ipv4-address = dec-octet "." dec-octet "." dec-octet "." dec-octet
ipv4-address = dec-octet dec-octet dec-octet dec-octet ; Incorrect: Missing period separators between octets.

dec-octet = DIGIT                 ; 0-9
          / 1*2DIGIT              ; 10-99
          / "1" 2DIGIT            ; 100-199
          / "2" %x30-34 DIGIT     ; 200-249
          / "25" %x30-35          ; 250-255
dec-octet = %x30-31 DIGIT ; Incorrect: Restricts the first digit of `dec-octet` to 0 or 1 only.

; Terminal definitions
DIGIT = %x30-39 ; 0-9
```

Example 4: ABNF grammar for a simplified HTTP/1.1 request.

```
; HTTP Request Grammar
; This grammar defines the syntax of a simplified HTTP/1.1 request.

request = request-line *( header-field CRLF ) CRLF [ message-body ]

request-line = method SP request-target SP HTTP-version CRLF
; A request-line consists of the HTTP method, request target, and HTTP
version.
request-line = method request-target HTTP-version ; Incorrect: Missing
SP (space) delimiters.

method = token
; Method is one of the standard HTTP methods like "GET," "POST,"
"PUT," "DELETE," etc.

request-target = uri
; Request target can be a more specific definition based on your
needs.

HTTP-version = "HTTP/1.1"

header-field = field-name ":" OWS field-value OWS
; Header field consists of a field name and field value, separated by
a colon.

field-name = token
; Field name is a token.

field-value = *( field-content )
; Field value can contain field content.

field-content = field-vchar [ 1*( SP / HTAB ) field-vchar ]
; Field content consists of field v-characters optionally followed by
spaces or tabs.

field-vchar = VCHAR / obs-text ; Field v-characters can include
visible characters or extended characters.

message-body = *OCTET ; Message body can contain zero or more octets
(binary data).
```

```
message-body = OCTET ; Incorrect: Missing the "*" operator for repetition.

; Terminal definitions
SP = %x20 ; Space
HTAB = %x09 ; Horizontal tab
token = 1*<any CHAR except CTLs or separators>
VCHAR = %x21-7E ; Visible (printable) characters
obs-text = %x80-FF ; Characters not specified in the current standard
OCTET = %x00-FF ; Any 8-bit byte value
```

Example 5: ABNF grammar for SMTP basic commands.

```
; SMTP Grammar for Basic Commands
; This grammar defines the syntax of basic SMTP commands.

; Command to specify the sender's address
MAIL = "MAIL FROM:" SP "<" mailbox ">" CRLF
MAIL = "MAIL FROM:" "<" mailbox ">" ; Incorrect: Missing SP and CRLF

; Command to specify the recipient's address
RCPT = "RCPT TO:" SP "<" mailbox ">" CRLF

; Command to start the data transfer
DATA = "DATA" CRLF
DATA = "DATA" ; Incorrect: Missing CRLF

; Command to end the data transfer and send the message
QUIT = "QUIT" CRLF

mailbox = local-part "@" domain CRLF
local-part = 1*atext / quoted-string
domain = sub-domain *("." sub-domain) / "[" *dtext "]"
sub-domain u= atom
atext = ALPHA / DIGIT / "!" / "#" / "$" / "%" / "&" / "'" / "*" / "+"
/ "-" / "/" / "=" / "?" / "^" / "_" / "`" / "{" / "|" / "}" / "~"
atom = 1*atext
quoted-string = DQUOTE *qcontent DQUOTE
qcontent = qtext / quoted-pair
qtext = %x21 / %x23-5B / %x5D-7E ; Printable US-ASCII excluding "\"
and """
quoted-pair = "\" (VCHAR / WSP)
dtext = %x21-5A / %x5E-7E ; Printable US-ASCII excluding "[", "]", and
"\"

; Terminal definitions
SP = %x20 ; Space
CRLF = %x0D %x0A ; Carriage return and line feed
ALPHA = %x41-5A / %x61-7A ; Uppercase and lowercase letters A to Z
DIGIT = %x30-39 ; Digits 0 to 9
DQUOTE = %x22 ; Quotation mark
VCHAR = %x21-7E ; Visible (printable) characters
WSP = SP / HTAB ; White space
```