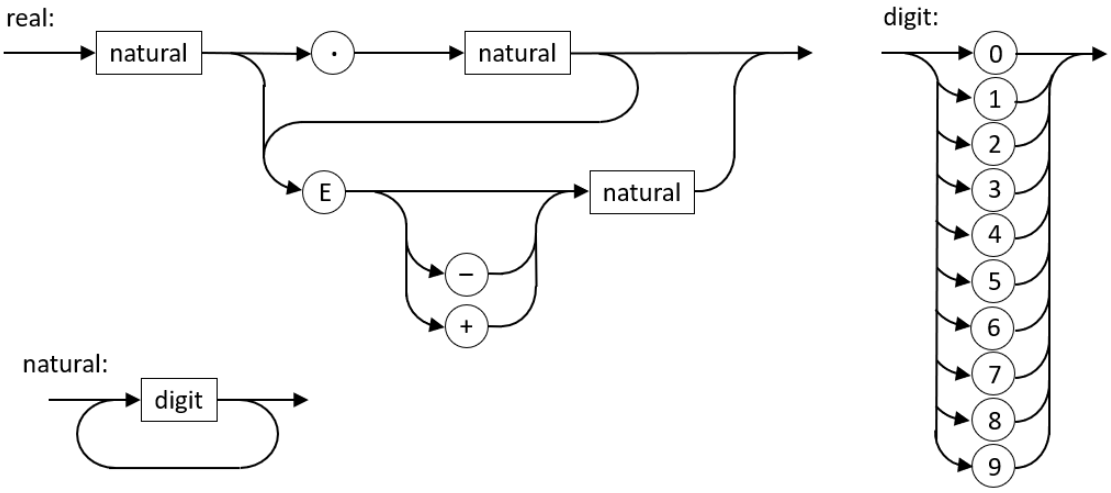
4.4.3 Backus-Naur Form

1) In a particular programming language, the correct syntax for a real number, natural number and digit is defined by the syntax diagrams in **Figure 1**.

Figure 1



a) Write Yes or No in the unshaded cells in Table 1 to identify whether or not the numbers listed in the table are valid real numbers which conform to the correct syntax for this language.

Table 1

Real number	Valid? (Yes/No)
36.012	**Yes**
54+31	**No**
12.48E+11	**Yes**

[3 marks]

b) In Backus-Naur Form (BNF) the following production rule has been written to define a digit:

<digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Write a BNF production rule to define a natural number that is equivalent to the definition in the syntax diagram in **Figure 1**.

[2 marks]

- **<natural> : : = <digit> | <digit><natural>**

2) A regular language is a language that can be defined by a regular expression.

a) Complete **Table 2** to show which of the statements about regular languages are true and which are false.

Table 2

Statement	True or False?
All regular languages can be represented using a finite state machine without outputs.	**True**
The set of strings defined by a regular language is always finite in size.	**False**
There are some languages which can be represented in Backus Naur Form (BNF) that are not regular languages.	**True**

[2 marks]

Figure 2 shows a set of Backus-Naur Form (BNF) rules that are used to define a language.

Figure 2

```
<sentence>    : : = <np><v>
<sentence>    : : = <v><np>
<np>          : : = <d><n>
<n>           : : = human | dog | cat | baby
<d>           : : = a | the
<v>           : : = ate | slept | drank | cuddle
<c>           : : = and | but | or
```

b) There are two rules in Figure 2 with on the left-hand side that are used to define what a valid sentence is in the language.

Write a single rule that could replace these two rules. Your new rule must not change what a valid sentence is.

[1 mark]

- **<sentence> : : = <np><v> | <v><np>**

c) Complete the cells of **Table 3** to show which of the strings are valid sentences for the language defined by the BNF rules in **Figure 2**.

Table 3

String	Valid sentence(Y/N)?
cuddle the cat	**Y**
drank a human	**Y**
the cat slept	**Y**
cat or dog	**N**

[1 mark]

d) The sentence *dog slept* is not currently valid in the language defined in **Figure 2**.

Change the language defined in **Figure 2** by either adding or modifying **exactly one rule** so that *dog slept* is a valid sentence.

You must not use the terminals *dog* or *slept* directly in the rule you write/change.

[1 mark]

- **Modify <np> : : = <d><n> | <n>**
- **// modify <sentence> : : = <np><v> | <n><v>**
- **// modify < sentence > : : = <v><np> | <n><v>**
- **// modify < sentence > : : = <np><v> | <v><np> | <n><v>**
- **// new rule < sentence > : : = <n><v>**
- **// new rule <np> : : = <n>**

The sentence *the cat slept but the dog drank* is not currently valid in the language defined in **Figure 2**.

To allow this to be a valid sentence either one of the following two additional BNF rules could be used.

<sentence> : : = <np><v><c><np><v>
<sentence> : : = <sentence><c><sentence>

e) State the number of different sentences defined by the rule:

<sentence> : : = <np><v><c><np><v>

You should only use the rules in **Figure 2** when working out your answer to this question. **Do not** include any additional sentences that your answer to question **d)** would make valid.

You can get full marks for this question for stating either the number of valid sentences or the full calculation needed to work out the number of valid sentences.

You should show your working.

[1 mark]

- **8x4x3x8x4 // 2x4x4x3x2x4x4**
- **3072**

f) State how many more sentences are defined by the rule:

<sentence> : : = <sentence><c><sentence>

than by the rule:

<sentence> : : = <np><v><c><np><v>

[1 mark]

- **Infinitely more**

3) A regular language is one that can be defined using a regular expression.

Figure 3 shows definitions for six different languages.

Figure 3

Language A

<string> : : = <term><op><term>
<string> : : = (<string>)
<op> : : = + | -
<term> : : = a | b

Language B

(a|b)+ab*

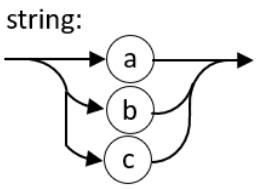
Language C

<string> : : = a | b | <num>
<string> : : = <string> a | <string> b
<digit> : : = 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<num> : : = <digit><num> | <digit>

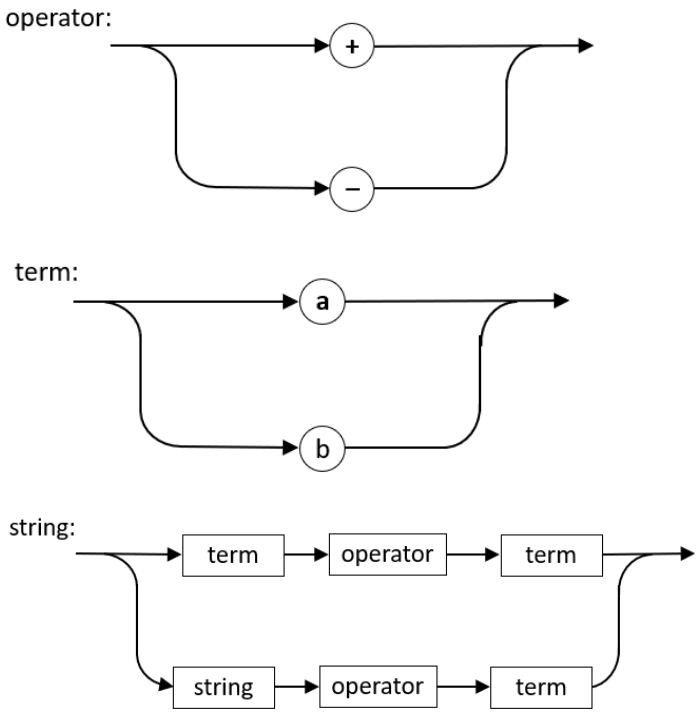
Language D

$\{a^n b \mid n \geq 1\}$

Language E



Language F



a) Complete **Table 4** to show which of the languages in **Figure 3** are regular languages.

Table 4

Language	Regular language (Y/N)?
Language A	**N**
Language B	**Y**
Language C	**Y**
Language D	**Y**
Language E	**Y**
Language F	**Y**

[3 marks]

b) Show that the language defined by the union of the sets {bn | n > 0} and {a, ab} is a regular language by writing a regular expression for the language.

[2 marks]

- **a|ab|b+ // ab?|b+**

4) Backus-Naur Form (BNF) can be used to define the rules of a language.

Figure 4 shows an attempt to write a set of BNF production rules to define a language of full names.

Figure 4

Note: underscores (_) have been used to denote spaces.

Note: rule numbers have been included but are not part of the BNF rules.

Rule number	
1	<fullname> : : = <title>_<name>_<endtitle> | <name> | <title>_<name> | <name>_<endtitle>
2	<title> : : = MRS | MS | MISS | MR | DR | SIR
3	<endtitle> : : = ESQUIRE | OBE | CBE
4	<name> : : = <word> | <name>_<word>
5	<word> : : = <char><word>
6	<char> : : = A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z

BNF can be used to define languages that are not possible to define using regular expressions. The language defined in **Figure 4** could not have been defined using regular expressions.

a) Complete **Table 5** below.

Table 5

Rule number (given in Figure 4)	Could be defined using a regular expression (Y/N)?
1	**Y**
2	**Y**
3	**Y**
4	**N**
5	**N**
6	**Y**

[1 mark]

There is an error in rule 5 in Figure 2 which means that no names are defined by the language.

b) Explain what is wrong with the production rule and rewrite the production rule so that the language does define some names – the names 'BEN D JONES', 'JO GOLOMBEK' and 'ALULIM' should all be defined.

[2 marks]

- **There is no non-recursive/base case;**
- **<word> : : = <char><word> | <char>**