



Object code characteristics

What is object code?

- Code produced by a compiler or assembler
 - Code produced by an interpreter
 - Code produced when using the object-oriented programming paradigm
 - Another name for a program produced in a low-level language
-
-
-

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





Compiler characteristics

High-level languages need to be translated. This can be done by either a **compiler** or an **interpreter**.

Select the **three** statements that are features of a compiler.

- Errors are discovered line by line, so can be corrected as they are detected.
 - Code is translated line by line while execution is underway.
 - The source code must be given to the user.
 - Any errors are reported at the end of the process.
 - All code is translated prior to being executed.
 - Only an executable file is given to the user, not the original source code.
-
-
-

Quiz:

[STEM SMART Computer Science Week 37](#)

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





Stages of translation 1

Complete the sentence by dragging and dropping the words provided into the correct position.

Programmers write [] code. [] code is executed by a virtual machine.
Compilers produce [] code. A processor can only run [] code.

Items:

[byte](#) [executable](#) [object](#) [source](#)

Quiz:

[STEM SMART Computer Science Week 37](#)

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





Creating executable code

There are many stages in the process of creating an executable file. Put the following stages into the correct order.

Available items

object code

compiler

source code

executable file

linker

Quiz:

[STEM SMART Computer Science Week 37](#)

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





Lexical analysis

During lexical analysis, a compiler is tokenising the following lines of code:

Pseudocode

```
1 user = "mathematician"  
2 pi = 3.142  
3 OUTPUT ("Pi = ", pi)
```

A list of tokens used in the process is:

- identifier
- literal
- leftpar
- rightpar
- quote
- number
- assignment
- comma
- keyword

The tokens used for the first line of code are: identifier, assignment, quote, literal, quote

What is the correct **order** of tokens that the compiler produces for the second line and third lines of code?

Drag and drop the tokens into the spaces provided. You **will need** to use some tokens more than once.

Drag and drop the tokens into the spaces provided. You will need to use some tokens more than once.

Items:

keyword leftpar number assignment quote rightpar comma
identifier literal

Quiz:



Bytecode description

The following text passage provides a description of **bytecode**. Some of the terms are missing.

Complete the passage by dragging and dropping the terms into the correct place.

Bytecode is created when [] code is partially translated using an [].
Bytecode is further [] and [] line by line by a []. Use of
bytecode improves performance while allowing platform independence.

Items:

Quiz:

[STEM SMART Computer Science Week 37](#)

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





Interpret RPN

An expression has been written using **infix** notation.

$$3 \times (10 + 5)$$

Convert the expression to **Reverse Polish Notation** (postfix) so that the two expressions evaluate to the same result.

Answer: [] [] [] [] [] [] [] [] [] []

Items:

- 3
 - ×
 - 5
 - +
 - 10
-
-
-

Quiz:

[STEM SMART Computer Science Week 37](#)

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





Convert expression from infix to postfix

How would the infix expression $3 + 4 * 2 - 1$ be represented in Reverse Polish notation (postfix)?

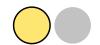
- 3 4 + 2 * 1 -
 - 3 4 * 2 + 1 -
 - 3 4 2 + * 1 -
 - 3 4 2 * + 1 -
-
-
-

Quiz:

[STEM SMART Computer Science Week 37](#)

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





Interpret BNF

The following set of BNF production rules can be used to define the term *calculation*:

```
calculation ::= <number><symbol><number>
number ::= <sign><real>|<real>|<sign><integer>|<integer>
integer ::= <digit>|<integer><digit>
real ::= <integer>.<integer>
digit ::= 0|1|2|3|4|5|6|7|8|9
symbol ::= +|-|*|/|^-|**
sign ::= +|-
```

Which of the following is **not** a valid *calculation* according to the rules specified?

- 23**3.5
 - +23//3.5
 - 23^-+3.5
 - 23.5++3.5
-
-
-

Quiz:

[STEM SMART Computer Science Week 37](#)

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





BNF base case

Two BNF rules have been written as follows:

```
<word> ::= <letter> | <word><letter>
```

```
<letter> ::= 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
```

The rule for word is recursive. When is the base case reached?

- When there is only one letter remaining
 - When there are no letters remaining
 - When the only letter that remains is Z
 - When the only letter that remains is A
-
-
-

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.

