1.3.6 Languages & Translators

1.3.7 High- and low-level languages and their translators

Revision Checklist

Learning Outcome	To Read	Have Read	To Revise	Have Revised	Prepared
1.3.7: High- and low-level languages and their translators					
Show understanding of the need for both high-level and low-level languages					
Show understanding of the need for compilers when translating programs written in a high-level language					
Show understanding of the use of interpreters with high-level language programs					
Need for assemblers when translating programs written in assembly language					
Solving past paper questions					

Programming Language:

"A set of rules that provide a way of telling computer what operation to perform is called a Programming Language."

A **COMPUTER PROGRAM** is a list of instructions that enable a computer to perform a specific task. Computer needs instructions to perform a task.

There are some rules and regulations to write programs and these rules and regulations are programming languages.

Different types of programming languages are given below:

1. Low Level Language (Machine Language and Assembly Language)
2. High level Languages

1) Machine Language: (Low Level Language)

Machine languages are the most basic languages. They consist of a series of current signals 'On' and 'Off'. On is represented by '1' and off is represented by '0'. It is 1st generation of language.

Machine Language Code
00010010111101010100111110101010 01000100101000011111010101011111 01010001111101010111011001001100 00010101001010101011111011111111

Machine language is also called Binary Digits. It is computer's language in which different parts of computer communicate with each other. 0s and 1s are called bit (binary digit).

Machine Language Code
12EF A243 4EC2 33C2 3D23 4E23 4FFA CA12 ACEF F234 DE12 F98A

Machine language is easy for computer and difficult for programmer. Every computer has its own unique machine language. It means machine for Apple Macintosh is different from IBM compatible.

2) Assembly Language: (Low Level Language)

Assembly language consists of English like symbolic codes known as mnemonics. It is a second generation of language.

They represent common strings of machine codes. A language translator, Assembler, is used to convert source code of Assembly language program into object code of machine language. Though Assembly languages are easier than machine language but they are highly detailed and cryptic. So programmers seldom write programs in Assembly language.

Assembly Code (Source code)
LDD 101 ADD 102 STO 103

Advantages of Assembly languages:

- to make use of special hardware
- to make use of special machine-dependent instructions
- to write code that doesn't take up much space in primary memory
- to write code that performs a task very quickly.

In order to understand this program the programmer needs to know that

- **LDD** means load the value of the variable from memory into the accumulator
- **ADD** means add the value of another variable to the value stored in the accumulator
- **STO** means store the value for accumulator into memory.

3) High Level Languages

High-level languages consist of familiar English words as result programmers can read, write and understand programs easily.

A language translator (compiler or interpreter) translates source code of High Level Language into object code. That's why the program of High Level Language can be executed on any machine.

High Level Code (Source code)
For Count=1 to 10 Print "Allah" Next Count

High-level languages are designed with programmers in mind; programming statements are easier to understand than those written in a low-level language. This means that programs written in a high-level language are easier to:

- read and understand as the language used is closer to human language
- write in a shorter time
- debug at the development stage
- maintain once in use.

Examples: COBOL, Pascal, FORTRAN, GWBASIC, C++, JAVA, VB.Net.

Source Code:

"A program written in any language except the machine language is called Source Code."

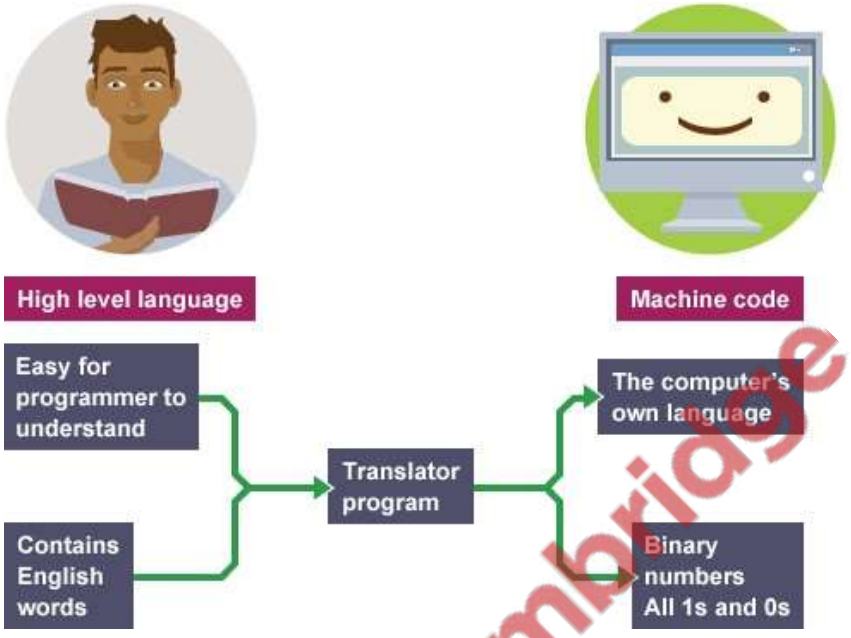
Programmers can write programs in Machine Language, Assembly Language and High Level Languages like COBOL, PASCAL, and GWBASIC etc. The programs, written in Assembly Language or any High Level Language are called Source Code.

Object Code:

"A program written in Machine Language is called Object Code."

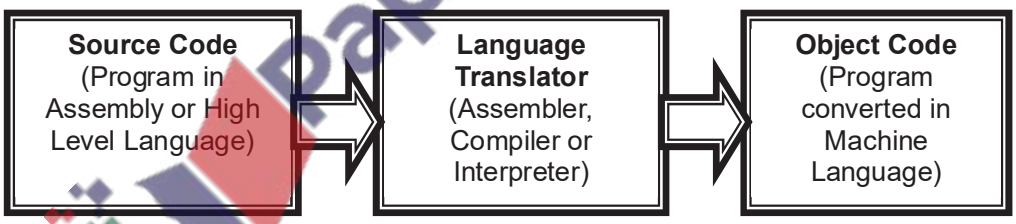
Computer can understand only machine language code. Language translators converts source code of High Level Language into machine code, these converted codes are Object Codes.

Language Translator:



"Language Translators are programs that convert or translate the instructions in Assembly Language or High Level Language i.e. source code into instructions of machine language i.e. object codes are called Language Translators."

Computer can understand only machine language. Programs written in Assembly Language and High Level Language are translated into Machine Language so as computer can understand execute them.

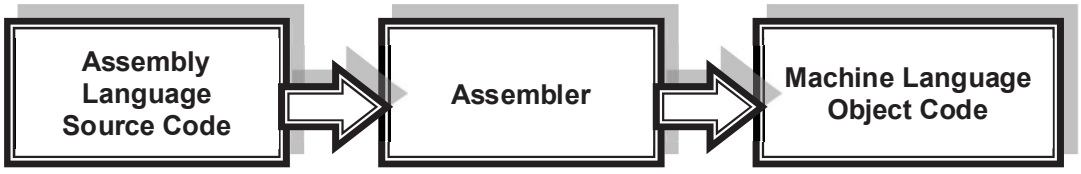


Following are three types of language translators:

1 Assembler:-

"Assembler is a language translator that translates source program written in Assembly Language into object code in machine language."

Computer can understand only machine language. Assemblers translate programs written in Assembly Language into Machine Language so as computer can understand execute them.



2 Interpreters:-

"An interpreter is a language translator that translates source program written in High Level Language into object code in machine language during step by step execution of program."

Computer can only understand instruction written in machine language. Interpreters translate one by one instruction of High Level Language source code into object code of Machine Language.

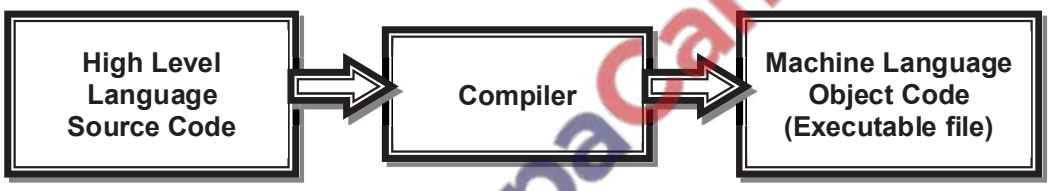
Interpreter translates one instruction which is executed before translation of next instruction. The object code is not saved so the source code is interpreted every time before the execution.

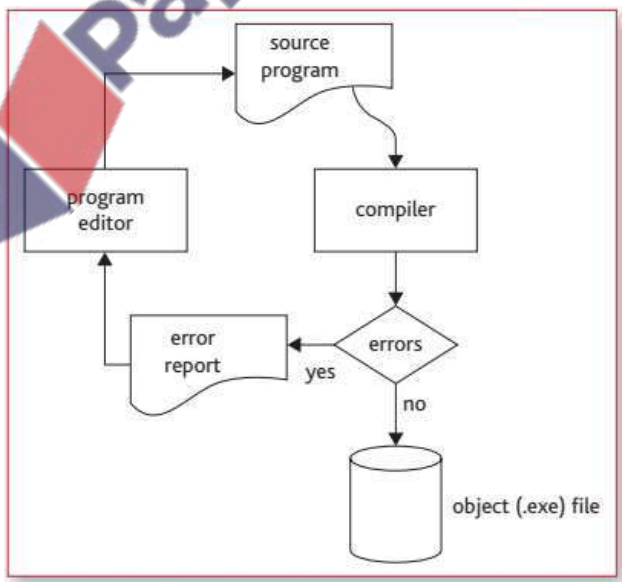
Each high level language has its own language translator.

3 Compilers:-

"A compiler is a language translator that translates whole source program written in High Level Language into object code in machine language before execution of program."

Compilers translate all instructions before executions. The object code is saved as object file, which is executable. Object files don't need compiler or language for execution.





Compiler	Interpreter	Assembler
Translates a high-level language program into machine code.	Executes a high-level language program a statement at a time.	Translates a low-level language program into machine code.
An executable file of machine code is produced.	No executable file of machine code is produced.	An executable file of machine code is produced.
One high-level language statement can be translated into several machine code instructions.	One high-level language program statement may require several machine code instructions to be executed.	One low-level language statement is usually translated into one machine code instruction.
Compiled programs are used without the compiler.	Interpreted programs cannot be used without the interpreter.	Assembled programs can be used without the assembler.
A compiled program is usually distributed for general use.	An interpreter is often used when a program is being developed.	An assembled program is usually distributed for general use.

Syntax errors

An error due to which, a program can't be executed is known as syntax error.

When a program is being compiled, if any syntax errors are found no translated program is produced. Instead, a list of all the errors in the whole program is produced.

When a program is being interpreted, the interpreter performs the actions specified by each statement until a syntax error is found.

Logic errors

Logical errors are the errors which don't stop execution of program but the program don't produced required result.

Run-time error

An ***error*** that occurs during the execution of a program.

For example, Divide-by-zero error or **running** out of memory will often cause a **runtime error.**

Integrated/Interactive Development Environment (IDE)

Most high-level programming languages offer the use of an IDE for program development. This contains an editor with an interpreter and/or compiler together with debugging tools, which can improve the speed of program development.

Exam Style Questions:

Q1) A syntax error can occur when writing a program.

(a) State what is meant by a syntax error, giving an example.

.....

.....

.....

..... **[2]**

(b) Describe tools and facilities available in an integrated development environment (IDE) which can help the programmer to identify and correct syntax errors.

.....

.....

.....

.....

.....

.....

.....

..... **[4]**

Expected Answer	Mark	Additional Guidance
• An error in the rules/grammar of the language • Any suitable example	[2]	"A spelling error" is acceptable as an example but not as a definition of syntax error. So e.g. "A spelling error such as ED IF instead END IF" is worth 1 mark only.
• Error messages/translator diagnostics • Produced when translating/by the compiler • ... or on the fly while writing code • Attempts to tell you what the error is • And indicate where the error is/line numbers/underlines • Editor... • ... allows you to enter the corrected code	[4]	Translator includes compiler/interpreter

Topical Questions from Past Papers

Q 1) Summer 2015 P11

10 Five statements about interpreters and compilers are shown in the table below.
Study each statement.
Tick (✓) to show whether the statement refers to an interpreter or to a compiler. [5]

Statement	Interpreter	Compiler
takes one statement at a time and executes it		
generates an error report at the end of translation of the whole program		
stops the translation process as soon as the first error is encountered		
slow speed of execution of program loops		
translates the entire program in one go		

Examiner's Comments on Question 10

Candidates normally gained full marks or no marks for this question. They could often identify the correct statements but sometimes confused the role of an interpreter and a compiler.

Q 2) Summer 2015 P12

9 (a) Five statements about interpreters and compilers are shown in the table below.
Study each statement.
Tick (✓) to show whether the statement refers to an interpreter or to a compiler.

Statement	Interpreter	Compiler
creates an executable file that runs directly on the computer		
more likely to crash the computer since the machine code produced runs directly on the processor		
easier to debug since each line of code is analysed and checked before being executed		
slow speed of execution of program loops		
it is more difficult to modify the executable code, since it is in machine code format		

(b) State why a compiler or an interpreter is needed when running a high-level program on a computer.

.....
.....
.....[1]

(c) Give **one** benefit of writing a program in a high-level language.

.....

.....

..... [1]

(d) Give **one** benefit of writing a program in a low-level language.

.....

.....

..... [1]

(e) Study the following three sections of code.

A:
```
1 0 1 0 1 1 0 1
1 1 0 0 1 1 1 0
1 0 1 1 0 1 1 1
```

B:
```
LDA X
INC X
STA Y
```

C:
```
FOR x ← 1 TO 10
READ n
ENDFOR
```

Identify, using the letters A, B or C, which of the above codes is an example of assembly code, high-level language code or machine code:

Assembly code .....

High-level language code .....

Machine code .....[2]

Examiner's comments on Question 9(a), 9(b), 9(c), 9(d) and 9(e)

In part (a) many candidates demonstrated a good knowledge of compilers and interpreters, gaining full marks. The most common errors for incorrect answers occurred in the first and fourth rows.
In parts (b), (c) and (d) candidates either gave a clear description demonstrating a good level of knowledge or a very vague response. Common errors were a reference to machine language rather than machine code in part (b) and not stating who machine code was easier to understand for in part (c). In part (d) a number of candidates referred to the amount of memory used and it running faster, but with modern processors this is often not the case, or is very minimal, so not a true advantage.
In part (e) most candidates gave a correct response.

Q 3) Summer 2016 P12

1 Complete the following by writing either *compiler*, *interpreter* or *assembler* in the spaces provided. [3]

..... – translates source code into object code.

..... – translates low-level language into machine code.

..... – stops the execution of a program as soon as it encounters an error.

Examiner Report Question 1

Most candidates answered this question accurately. The most common error was confusing the role of a compiler and an interpreter.

Q 4) Winter 2016 P12

1 (a) Give **two** reasons why a programmer would choose to write code in a low-level language.

1 .....

.....

.....

2 .....

.....

..... [2]

(b) High-level languages require either an interpreter or a compiler to translate the program.

The table below lists a number of statements about language translators.

Tick (3) to show which statements refer to interpreters and which refer to compilers. [5]

Statements	Interpreter	Compiler
Translates the source code into machine code all at once		
Produces an executable file in machine code		
Executes a high-level language program one instruction at a time		
Once translated, the translator does not need to be present for the program to run		
An executable file is produced		

Examiner Report

In part **(a)** few candidates gained marks for this question, demonstrating a lack of understanding of what the benefits are of a low-level language. Some candidates made a vague reference to the program taking up less space, but this was not detailed enough for a mark; reference to memory needed to be present.

In part **(b)** the full range of marks was seen from candidates. The most common error was candidates confusing the role of a compiler and an interpreter.

Q 5) Winter 2016 P11& 13

6 High-level or low-level languages can be used when writing a computer program.

State **two** advantages of using a high-level language and **two** advantages of using a low-level language.

High-level language advantage 1 .....

.....

High-level language advantage 2 .....

.....

Low-level language advantage 1 .....

.....

Low-level language advantage 2 .....

..... [4]

Examiner Report

Many candidates did not provide answers that gained marks for this question on high- and low-level languages. It was clear that candidates did not understand the difference between a high-level language and a low-level language, and the benefits of them. Answers were very vague for high-level language, citing it was easier, but without any reference, for example, easy to debug. Many candidates incorrectly stated that low-level language does not need any translation, that it is already machine code.

8 Four descriptions about compilers and interpreters are shown below.
Draw lines to indicate which descriptions refer to a compiler and which descriptions refer to an interpreter. [4]

Description		
It is more difficult to debug the code since one error can produce many other associated errors.		
The speed of execution of program loops is slower.		Compiler
It produces fast, executable code that runs directly on the processor.		Interpreter
It is easier to debug the code since an error is displayed as soon as it is found.		

Examiner Report
The full range of marks was seen from candidates in this question on compilers and interpreters. Some candidates only connected a single line from each translator. The question stated to draw lines, candidates should note this type of question means that there may be more than one line that can be drawn to connect terms to description. If a question states to draw a line, this is when only a single line should be drawn from each term to a description.

Q 6) March 2017 India

11 Three programmers are working on different projects:
• Alice is developing a program written in a low-level language
• Akbar is developing a program written in a high-level language
• Alex is preparing a program written in a high-level language for sale
State, with reasons, which type of translator each programmer should use. Each programmer should be using a different type of translator.

Alice .....
.....
.....
.....

Akbar .....
.....
.....
.....

Alex .....
.....
.....
..... [6]

Examiner Report
The full range of marks was awarded for this question. Common errors included incorrectly describing what the translation did rather than explaining why that translator was the most appropriate one to use.

Q 7) Summer 2017 P11

2 Programmers can use a high-level language to write a computer program.

(a) Explain what is meant by the term 'high-level language'.

.....
.....
.....
..... [2]

(b) A program written in a high-level language is translated into machine code. This is so that it can be processed by a computer.
Name one type of translator that can be used.

..... [1]

(c) Describe how your answer to **part (b)** translates this program.

.....
.....
.....
.....
.....
.....
..... [3]

Examiner's Comments Question 2(a)
Many candidates gained a mark by stating that high-level language is closer to English, some gained a further mark by stating an example of a high-level language. It would be beneficial for more candidates to demonstrate the knowledge that it is a language that is portable, being independent of any particular hardware.

Question 2(b)
Many candidates correctly stated that a compiler or an interpreter could be used. Some candidates

incorrectly stated that an assembler could be used. Candidates must understand that an assembler is not used to translate high level language.

Question 2(c)

Compiler

Most candidates correctly stated that a compiler translates the whole program as a single unit. Some candidates then stated that an executable file is produced. It would be beneficial for more candidates to demonstrate the knowledge that a report of errors is created and provided after the code is translated.
Some candidates were not accurate in their description and could not be awarded a mark, e.g. it translates all the program. This statement could be applied to any translator.

Interpreter

Most candidates stated that a compiler translates the program one line of code at a time. Some candidates then stated that the translator will stop when it finds an error in the code.
Some candidates were not accurate in their description and could not be awarded a mark, e.g. it translates itbit by bit. This statement requires more accuracy and candidates must demonstrate the knowledge it is translated a line of code at a time.

Q 8) Winter 2017 P13

10 Six statements about assembly language are shown.

Tick (✓)whether the statement is true or false. [6]

Statement	True(✓)	False(✓)
Assembly language uses mnemonic codes.		
Assembly language programs do not need a translator to be executed.		
Assembly language is a low-level programming language.		
Assembly language is specific to the computer hardware.		
Assembly language is machine code.		
Assembly language is often used to create drivers for hardware.		

Q 9) Summer 2018 P11

7 Translators, such as a compiler and an interpreter, are used when writing and running computer programs.

Describe how a compiler and an interpreter translates a computer program.

Compiler .....

.....

.....

.....

.....

.....

Interpreter .....

.....

.....

.....

.....

.....

..... [6]

Q 10) Summer 2018 P12

8 Dimitri is writing a computer program in a high-level language.

He needs to send just the machine code for the program to his friend, electronically. It is important that the program is executed as quickly as possible. Identify which translator will be most suitable f or Dimitri to use. Explain your choice.

Type of translator .....

Explanation .....

.....

.....

.....

.....

..... [4]

Q 11) Winter 2018 P12

6 (a) Many programmers write computer programs in high-level languages. The programs need to be translated into machine code to be read by the computer.

State **two** types of translator that can be used.

Translator 1 .....

Translator 2 ..... [2]

(b) Explain **two** reasons why a computer programmer may choose to write a program in a high level language, rather than a low-level language.

Reason 1 .....

.....

.....

.....

Reason 2 .....

.....

.....

..... [4]

(c) Three examples of computer code are given in the table.

Tick (✓) to show whether each example of computer code is **High-level language**, **Assembly language** or **Machine code**. [3]

Computer code	High-level language (✓)	Assembly language (✓)	Machine code (✓)
10110111 11001100 01011100			
FOR X = 1 TO 10 PRINT X NEXT X			
INP X STA X LDA Y			

Q 12) Winter 2018 P13

7 David is writing a program using a high-level language. The program will be published and sold for profit.

(a) David uses an interpreter when creating the computer program. State three features of an interpreter.

Feature 1 .....

.....

.....

Feature 2 .....

.....

.....

Feature 3 .....

.....

..... [3]

(b) David compiles the program when he has completed it. Explain two benefits of compiling the program.

Benefit 1 .....

.....

.....

.....

.....

Benefit 2 .....

.....

.....

.....

..... [4]

Q 13) March 2019 P12

4 Darius is writing a computer program that allows binary values to be calculated.

Darius chooses to write the program in a high-level language rather than a low-level language.

(a) Explain why Darius chooses to write the program in a high-level language.

.....

.....

.....

..... [2]

(b) Darius will use a translator to translate the program. He could use a compiler or an interpreter.

Five statements are given about compilers and interpreters.

Tick (✓) to show if the statement applies to a **Compiler** or an **Interpreter**. Statements may apply to both. [5]

Statement	Compiler (✓)	Interpreter (✓)
A report of errors is produced at the end of translation.		
The program is translated one line at a time.		
The program is translated from high-level language into machine code.		
An executable file is produced.		
The program will not run at all if an error is detected.		

Q 14) Summer 2019 P11

7 Annie writes a paragraph of text as an answer to an examination question about programming languages. Using the list given, complete Annie's answer by inserting the correct **six** missing terms. Not all terms will be used.

- Assembly
- Converter
- Denary
- Hexadecimal
- High-level language
- Low-level language
- Machine Code
- Source Code
- Syntax
- Translator

The structure of language statements in a computer program is called the ..... . A programming language that uses natural language statements is called a ..... . When programs are written in this type of language they need a ..... to convert them into ..... . A programming language that is written using mnemonic codes is called ..... language. This is an example of a ..... . [6]

Q 15) Winter 2019 P13

2 A programmer uses a high-level language to write a computer program. [4]

(a) Four statements are given about high-level programming languages. **Tick (✓)** to show if each statement is **True** or **False**.

Statement	True	False
High-level languages need to be translated into machine code to run on a computer		
High-level languages are written using mnemonic codes		
High-level languages are specific to the computer's hardware		
High-level languages are portable languages		

(b) Tick (✓) to show which of the following is an example of a high-level language program. [1]

Example program	Tick (✓)
1011100000110000 0000011011100010	
INP STA ONE INP STA TWO ADD ONE	

a = input()
b = input()
if a == b:
print("Correct")
else:
print("Incorrect")	

Q 16) March 20 P12

4 Assemblers, compilers and interpreters are types of translators. [5]

Tick (✓) to show which statements apply to each translator. Each statement may apply to more than one type of translator.

Statement	Assembler	Compiler	Interpreter
Translates low-level language to machine code			
Translates high-level language to machine code			
Produces error messages			
Translates high-level language one line at a time			
Produces an executable file			

Q 17) Summer 20 P12

2 Both an interpreter and a compiler can be used when writing a program in a high-level language.

(a) Explain why a programmer would make use of both an interpreter **and** a compiler.

.....

.....

.....

.....

.....

.....

..... [4]

(b) Give **three** reasons why a programmer would choose to write a program in a high-level language, instead of a low-level language.

Reason 1 .....

.....

Reason 2 .....

.....

Reason 3 .....

..... [3]

Q 18) 15a Summer 20 P11

9 Programs can be written in a low-level language.

(a) Identify **three** features of a low-level language.

Feature 1 .....

Feature 2 .....

Feature 3 ..... [3]

(b) Give **two** examples of a low-level language.

Example 1 .....

Example 2 ..... [2]

(c) Give **one** drawback of writing programs in a low-level language, instead of a high-level language.

.....

..... [1]

Q 19) March 20 P12

7 Adeel has used a high-level language to program a mobile application.

(a) Describe what is meant by a high-level language.

.....
.....
.....
..... [2]

(b) Adeel uses an interpreter while developing and testing the application.
Adeel uses a compiler when the application is ready to be shared with others.
Compare the features of interpreters and compilers.

.....
.....
.....
.....
.....
.....
.....
..... [4]

Marking Scheme

10 1 mark per correctly placed tick

statement	interpreter	compiler
takes one statement at a time and executes it	✓	
generates an error report at the end of translation of the whole program		✓
stops the translation process as soon as the first error is encountered	✓	
slow speed of execution of program loops	✓	
translates the entire program in one go		✓

Q 2) Summer 2015 P12

Page 10	Mark Scheme	Syllabus	Paper
	Cambridge O Level – May/June 2015	2210	12

9 (a) 1 mark per correctly placed tick

statement	interpreter	compiler
creates an executable file that runs directly on the computer		✓
more likely to crash the computer since the machine code produced runs directly on the processor		✓
easier to debug since each line of code is analysed and checked before being executed	✓	
slow speed of execution of program loops	✓	
it is more difficult to modify the code since the executable code is now in machine code format		✓

[5]

(b) Any **one** from:

- code is required to be converted into machine code/binary
- code needs to be produced that can be understood by the computer

[1]

(c) Any **one** from:

- close to English/native/human language
- easier/faster to correct errors/read/write
- works on many different machines/operating systems (portable)

[1]

(d) Any **one** from:

- work directly on registers/CPU
- more control over what happens in computer
- can use machine specific functions

[1]

(e) 1 mark per correct letter, maximum 2 marks

Assembly code: **B**
High-level language code: **C**
Machine code: **A**

[2]

Q 3) Summer 2016 P12

1 compiler

assembler

interpreter [3]

Q 4) Winter 2016 P12

1 **(a)** Any **two** from:

- direct access to computer processor / special hardware // machine dependent instructions
- uses up less memory
- can increase the speed of processing a program // executes instructions faster

[2]

(b)

Statements	Interpreter (✓)	Compiler (✓)
Translates the source code into machine code all at once		✓
Produces an executable file in machine code		✓
Executes a high-level language program one instruction at a time	✓	
Once translated, the translator does not need to be present for the program to run		✓
An executable file is produced		✓

[5]

Q 5) Winter 2016 P11& 13

6 Any **two** from:

High level language

- easier/faster to write code as uses English-like statements
- easier to modify as uses English-like statements
- easier to debug as uses English-like statements
- portable language code

Any **two** from:

Low level language

- can work directly on memory locations
- can be executed faster
- translated program requires less memory

[4]

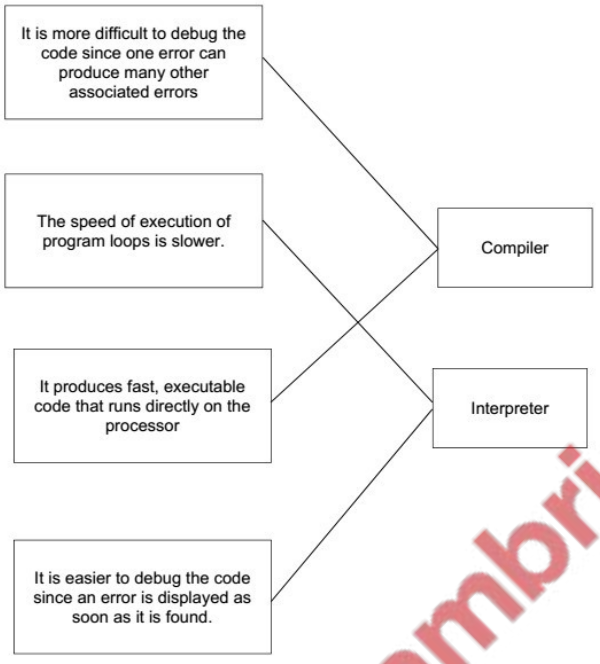
7 Any **four** from:

- reaches maximum brightness quickly
- colours are vivid
- good colour definition/contrast can be achieved
- screens can be thinner/thin
- more reliable as LED's are long lasting
- consume very little/less energy

[4]

Page 7	Mark Scheme	Syllabus	Paper
	Cambridge IGCSE – October/November 2016	0478	13

8



[4]

Q 6) March 2017 India

Question	Answer	Marks
11	Alice ∞ Assembler ∞ translates low level language into machine code / only option for low-level language programs Akbar ∞ Interpreter ∞ easy to identify where an error is / to debug a program Alex ∞ Compiler ∞ once translated a stand-alone program file is created / no need for the compiler when running the program	6

Q 7) Summer 2017 P11

Question	Answer	Marks
2(a)	**Two** from: ∞ Closer to human language // closer to English ∞ Independent of a particular type of computer/device/platform // portable language ∞ A language such as Python, Java, Pascal, etc. (any suitable example)	**2**
2(b)	**One** from: ∞ Compiler ∞ Interpreter	**1**
2(c)	Must relate to answer given in 2b. No follow through for incorrect answer in part 2b. **Compiler – Three** from: ∞ Translates the whole program as a complete unit / at once ∞ Creates an executable file / object code ∞ A report / list of errors in the code is created ∞ Optimises the source code (to run efficiently) **Interpreter – Three** from: ∞ Translates a program one line of code at a time ∞ Machine code is directly executed // The interpreter is used each time the program / code is executed ∞ Will identify an error as soon as it finds one in a line of code	**3**

Q 8) Winter 2017 P13

Question	Answer	Marks
10	1 mark for each correct tick	**6**

Statement	true (✓)	false (✓)
Assembly language uses mnemonic codes.	✓	
Assembly language programs do not need a translator to be executed.		✓
Assembly language is a low-level programming language.	✓	
Assembly language is specific to the computer hardware.	✓	
Assembly language is machine code.		✓
Assembly language is often used to create drivers for hardware.	✓	

Q 9) Summer 2018 P11

Question	Answer	Marks
7	Compiler Any **three** from: – Translates high-level language into **machine code/low level language** – Translates (the source code) all in one go/all at once – Produces an executable file – Produces an error report Interpreter Any **three** from: – Translates high-level language into **machine code/low level language** – Translates (the source code) line by line/statement by statement – Stops if it finds an error – Will only continue when error is fixed	**6**

Q 10) Summer 2018 P12

8	1 mark for correct translator, 3 marks for explanation: – Compiler Any **three** from: – Does not require recompilation // compiled program can be executed without a compiler … – ... therefore, allows faster execution – Provides an executable file … – … therefore, allows him to just send machine code – **Dimitri's friend** does not need translation/compilation **software** to execute the program	**4**

Q 11) Winter 2018 P12

Question	Answer	Marks
6(a)	∝ Compiler ∞ Interpreter	**2**
6(b)	**Four** from: ∞ Closer to human language/English … ∞ … so it is easier/quicker to read/write/understand ∞ … so it is easier/quicker to debug the program ∞ … therefore, less likely to make errors ∞ The program can be used on many different platforms … ∞ … because it is written in source code ∞ … because it is compiled into object code ∞ They have built-in functions/libraries … ∞ … this saves time when writing the program ∞ Do not need to manipulate memory addresses directly … ∞ … therefore, specialist knowledge of this is not required ∞ Only need to learn a single language … ∞ … as this can be used on many different computers	**4**

Question	Answer	Marks
6(c)	1 mark for each correct tick (✓)	**3**

Computer code	High-level language (✓)	Assembly language (✓)	Machine code (✓)

`10110111`
`11001100`
`01011100`
			✓

`FOR X = 1 TO 10`
`  PRINT X`
`NEXT X`
	✓		

`INP X`
`STA X`
`LDA Y`
		✓	

Q 12) Winter 2018 P13

Question	Answer	Marks
7(a)	**Three** from: ∞ It is a translator ∞ Translates (high level language) to low level language ∞ Executes one line at a time ∞ Translates source code line by line ∞ Runs error diagnostic ∞ Produces error messages to tell user location of error ∞ Stops execution when encounters errors ∞ Continues translating when an error is fixed	3
7(b)	**Four** from (Max three per benefit): ∞ Produces executable file … ∞ … this creates a smaller file size ∞ … more saleable ∞ Program will be machine independent / portable … ∞ … this means it can be used on any hardware ∞ No need for compiler to run executable file … ∞ … this means it will be quicker to run ∞ … customers can just execute the program ∞ Source code cannot be accessed … ∞ … therefore, code cannot be stolen / plagiarised	4
7(c)	**Three** from: ∞ Uses compression algorithm / by example e.g. RLE ∞ Repeating words / phrases / patterns identified… ∞ … replaced with value ∞ File / dictionary / index of phrases created ∞ Index will store word/phrase with value	3

Q 13) March 2019 P12

4(a)	**Two** from: – Closer to English statements / human language – Easier / quicker to write / read / understand / remember – Easier / quicker to debug – Less likely to make errors – One line of code can carry out multiple commands – Portable language	2
4(b)	1 mark for correct tick(s) for each statement	5
4(c)	– Lossy would remove data – Lossless does not remove data // No data can be lost … – Can be restored to original state … – … otherwise will not run / work correctly	4
4(d)(i)	– Sending device creates value from calculation on **data** // By example – Value is transmitted with the data – Receiving device performs same calculation – Values are compared **after transmission** // If values do not match … – … an error is detected	5
4d(ii)	– Parity check – Check digit – Automatic repeat request	3

Statement	Compiler	Interpreter
A report of errors is produced at the end of translation	✓	
The program is translated one line at a time		✓
The program is translated from high-level language into machine code	✓	✓
An executable file is produced	✓	
The program will not run at all if an error is detected	✓	

Q 14) Summer 2019 P11

7	1 mark for each correct term, in the correct place: • Syntax • High-level language • Translator • Machine code • Assembly • Low-level language	6

Q 15) Winter 2019 P13

Question	Answer	Marks
2(a)	**One** mark for each correct row	4

Statement	True (✓)	False (✓)
High-level languages need to be translated into machine code to run on a computer	✓	
High-level languages are written using mnemonic codes		✓
High-level languages are specific to the computer's hardware		✓
High-level languages are portable languages	✓	

2210/13 Cambridge O Level – Mark Scheme PUBLISHED October/November 2019

Question	Answer	Marks
2(b)	**One** mark for the correct tick	1

Example program	Tick (✓)
`1011100000110000` `0000011011100010`	

`INP`
`STA ONE`
`INP`
`STA TWO`
`ADD ONE`	

`a = input()`
`b = input()`
`if a == b:`
`  print("Correct")`
`else:`
`  print("Incorrect")`
	✓

Q 16) March 20 P12

Question	Answer	Mark
4		5

Statement	Assembler (✓)	Compiler (✓)	Interpreter (✓)
Translates low-level language to machine code	✓		
Translates high-level language to machine code		(✓)	✓
Produces error messages	(✓)	✓	✓
Translates high-level language one line at a time			✓
Produces an executable file	(✓)	✓	

1 mark per each correct row:

NOTE: tick shown in brackets (✓) is optional

Q 17) Summer 20 P12

Question	Answer	Marks
2(a)	Any **four** from: – To translate the high-level language into low-level language – Interpreter used whilst writing the program – Interpreter used to debug code line by line – Compiler used when program completed – Compiler used to create separate executable file (so compiler no longer needed) – If it runs first time in a compiler there are no syntax errors	4
2(b)	Any **three** from: – Easier to understand // Don't know assembly code – Easier to debug – Easier to maintain – Portable – Knowledge of manipulating memory locations/registers not required – Can use an IDE – Greater range of languages	3

Q 18) 15a Summer 20 P11

9(a)	Any **three** from: – Closer to/is machine code – May use mnemonics – May need an assembler to be translated – One line of code represents a single instruction – Machine dependent – Have direct access to memory locations/registers	3
9(b)	– Assembly code – Machine code	2
9(c)	Any **one** from: – It is more difficult to understand – Error prone – Have to manipulate memory locations – Machine dependent	1

Q 19) March 20 P12

7(a)	Any **two** from: – Makes use of words // close to human language – Machine independent // portable – Problem / logic focussed – Needs to be translated/interpreter/compiled (to low-level for processing by computer) // needs converting to machine code	**2**
7(b)	**Four** from **Max 2** for only giving compiler/interpreter features – **Both** translate high level / source code to machine code – **Both** generate error diagnostics/messages // identify errors – Interpreter translates one line at a time // checks one line and then runs it – Compiler translates whole code in one go // checks all code and then runs it – Interpreter **stops** when meets an error – …and then allows you to continue running from where you stopped // correct errors in real-time – Compiler provides list of all errors – Interpreter does not produce an **executable** file – Compiler produces an **executable** file	**4**