CATEGORIES AND CLASSIFICATION OF PROGRAMMING LANGUAGES

11.1B Programming paradigms

LEARNING OBJECTIVES:

- distinguish between generations of programming languages
- classify programming languages into low and high-level
- Analyze the advantages and disadvantages of high-level languages
- Analyze the advantages and disadvantages of low-level languages

SUCCESS CRITERIA:

Knowledge

- Name generation of programming language;
- Give differences between HLL&LLL;

Comprehension

- Explain using LLL;
- State differences between machine code & assembler;

Analysis

- Define the level of the code programming languages.

Glossary

ENGLISH	РУССКИЙ
High level language	Язык высокого уровня
Low level language	Язык низкого уровня
Machine code	Машинный код
Assembler	Ассемблер
Generation	Поколение
Execute	Выполнять
Binary	Двоичный код

DISCUSSION

- What are programming languages?
- Which PL you have experience in?
- What are the differences between Object-oriented and structured PL?

SEARCH INFORMATION AND MAKE A online presentation

1 group	2 group
1. How many generations of PL exist? 2. What programming languages are considered to be of high level? 3. When (year) did PL start first to be developed? 4. When (year) were the 1,2 generation programming languages created? 5. Find examples for 1-2 generation of PL	1. Which PL are considered to be the 3rd generation, the 4th generation. 2. Who is the first programmer? 3. When (year) were the 3,4,5 generation programming languages created? 4. Find examples for 3-5 generation of PL

Links to online presentations

Yerzhanova Aizhan

https://docs.google.com/presentation/d/1UZ0u5tFrBs3pkmPgFv52PUvuFghNuqR5jvr-qOHYrWk/edit#slide=id.g6158184925_0_46

https://docs.google.com/presentation/d/1zROIL7CZo8c9HvfJiZufKgmeF-VbRnCeXAx7XdFp0Is/edit#slide=id.p

https://docs.google.com/presentation/d/1NrBfCwApGVU9mykFnxeAK4mZclkNXMQbOhbUGQ595Og/edit#slide=id.p

Presentation time 4 minutes

Generations of PL	Period (Year)	Names of programming languages	Low/High level language
1th generation			
2th generation			
3th generation			
4th generation			
5th generation			

Activity 1
Fill in the table

Potentially fast programs / Programming language C / Difficult to read by human/ Similar to English / Less difficult to learn / Have a friendly interface / More difficult to modify and maintain / More difficult to learn / Assembler / Must be translated / Easier for computer to read / Converting requires extra time / Easier to read for man / Easily understood by hardware / Pascal / Machine code

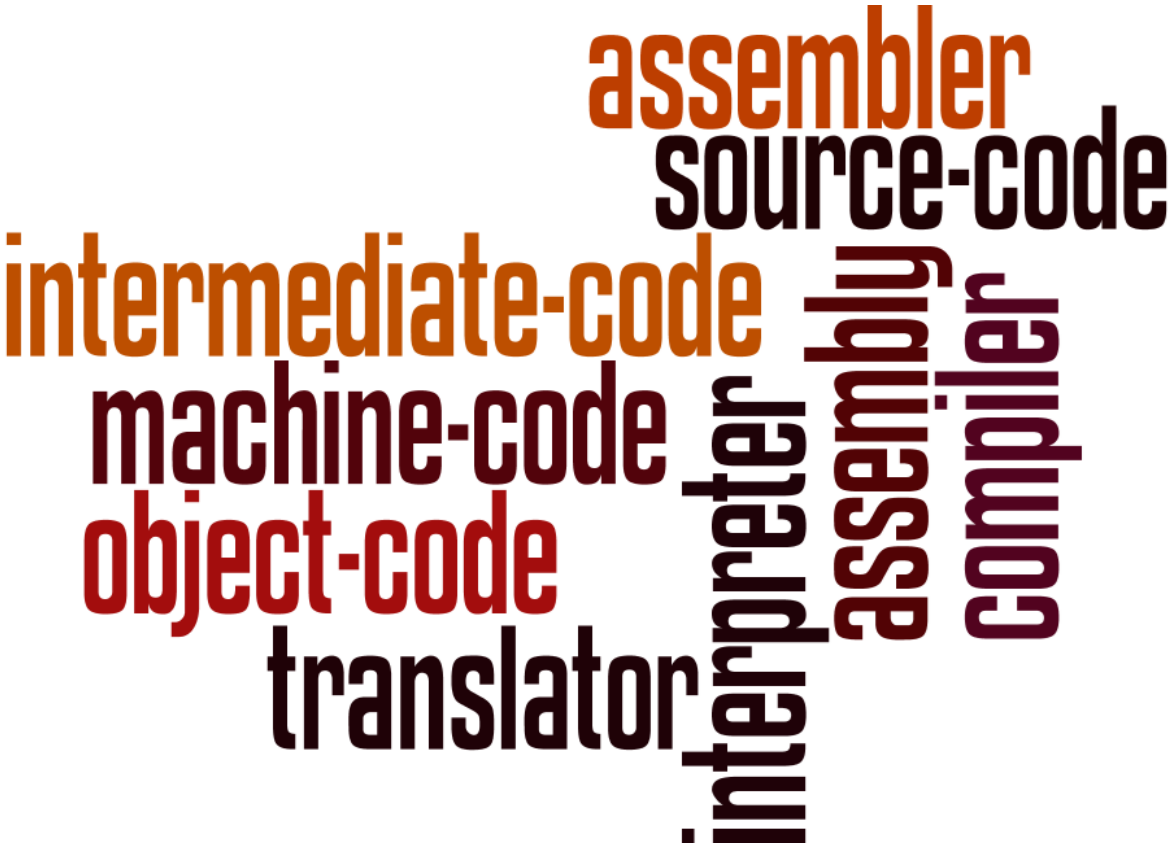
High Level PL	Low level PL

Fill in the table (answer)

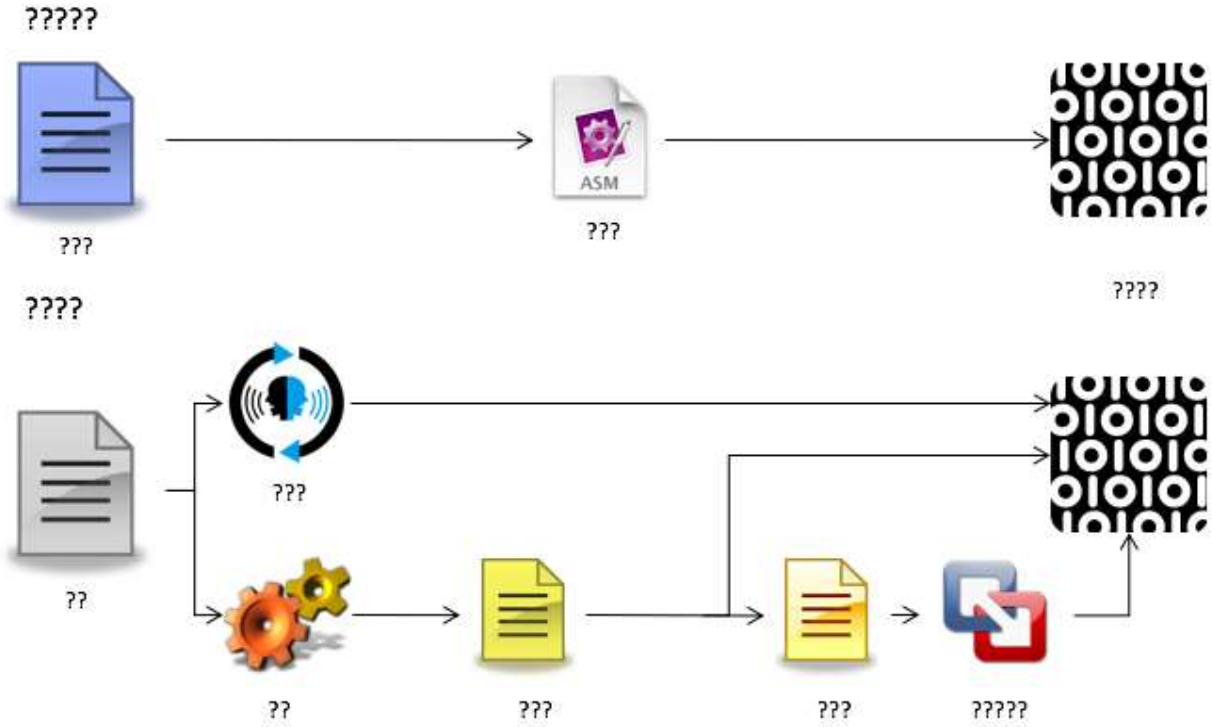
High Level PL	Low level PL
Programming language C Similar to English Less difficult to learn Have a friendly interface Must be translated Converting requires extra time Easier to read for man Pascal	Potentially fast programs Difficult to read by man More difficult to modify and maintain More difficult to learn Assembler Easier for computer to read Easily understood by hardware Machine code

Activity 2

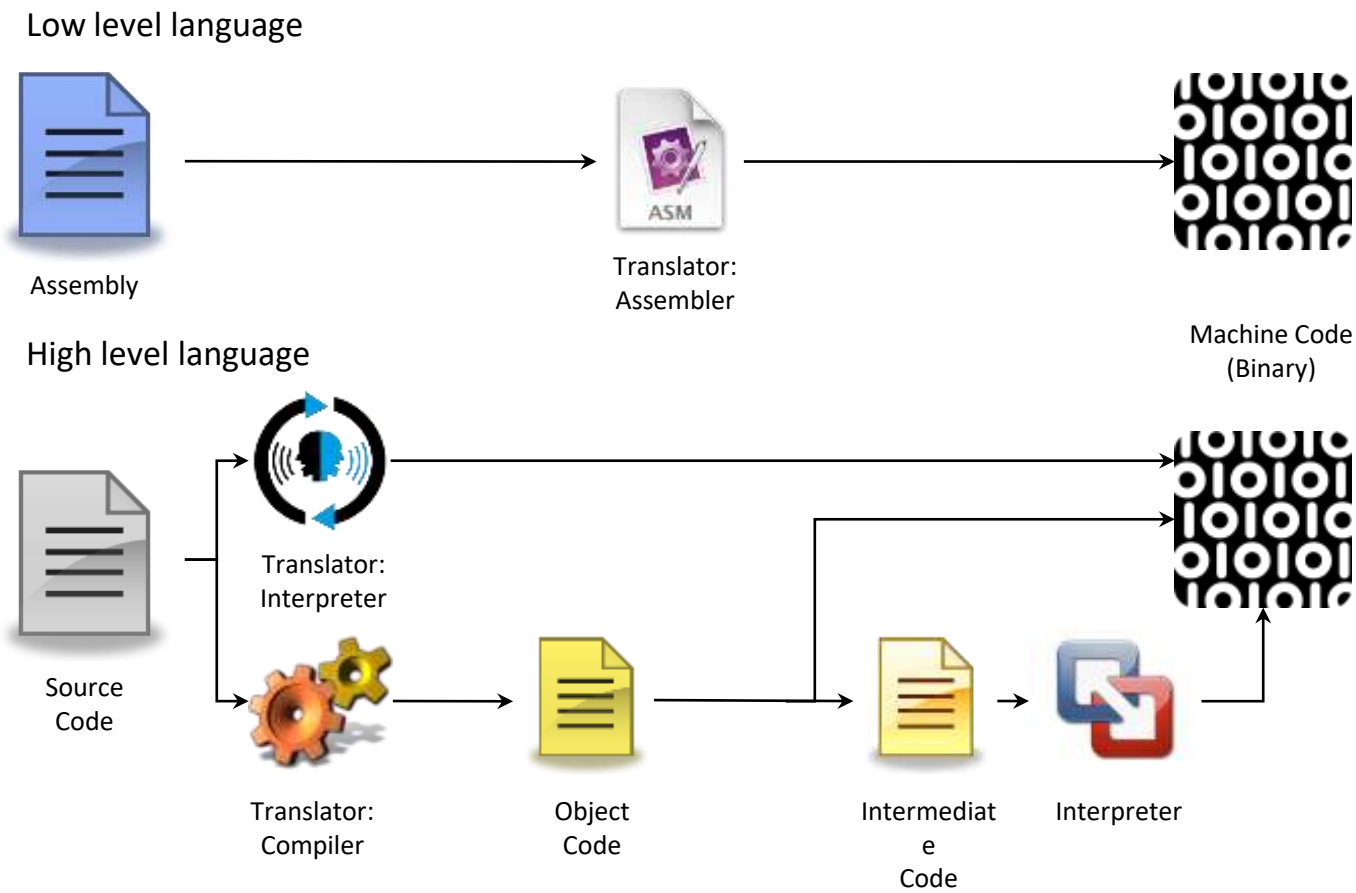
Using all the words in this word-wall, create a diagram to show how all the concepts are linked together.

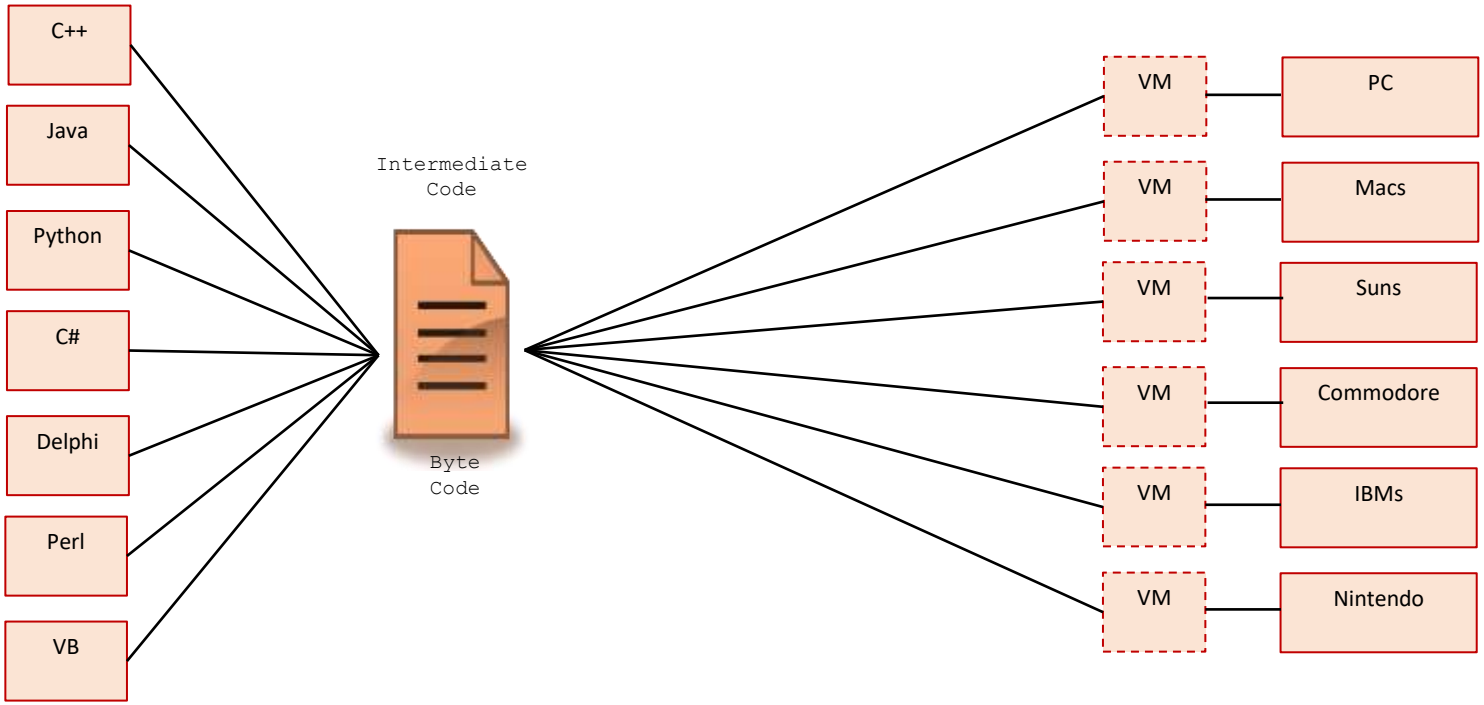


Assessment criteria
Understand the role of each of the following:
• assembler • compiler • interpreter.
Explain the differences between compilation and interpretation. Describe situations in which each would be appropriate.


?????
???
ASM
???
????
????
???
??
??
???
???
?????

Answer


Low level language
Assembly
ASM
Translator:
Assembler
Machine Code
(Binary)
High level language
Translator:
Interpreter
Source
Code
Translator:
Compiler
Object
Code
Intermediat
e
Code
Interpreter

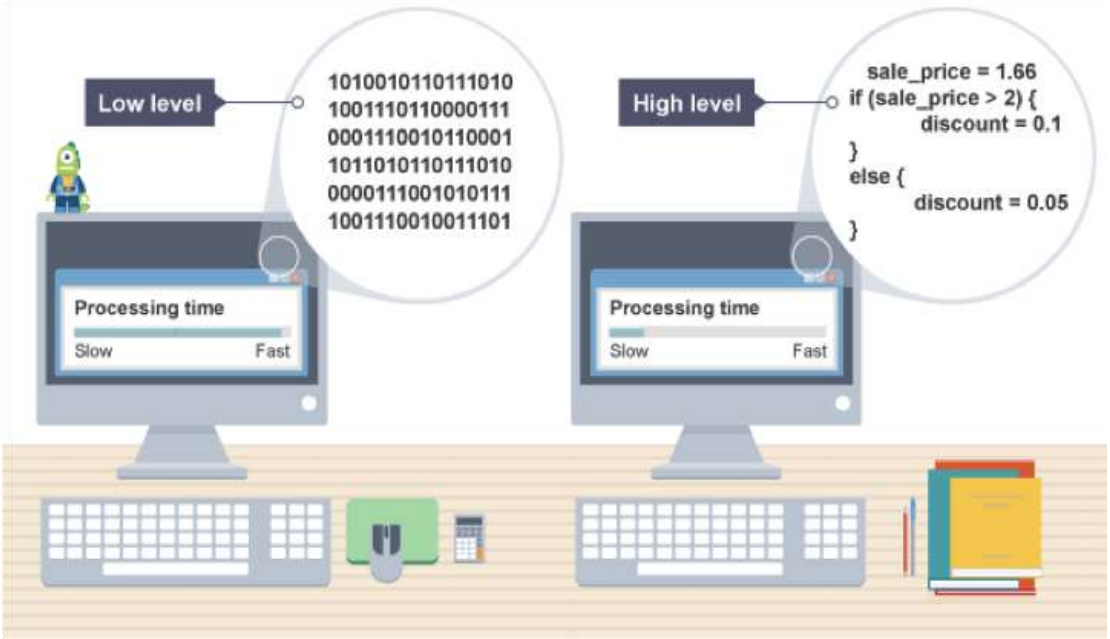


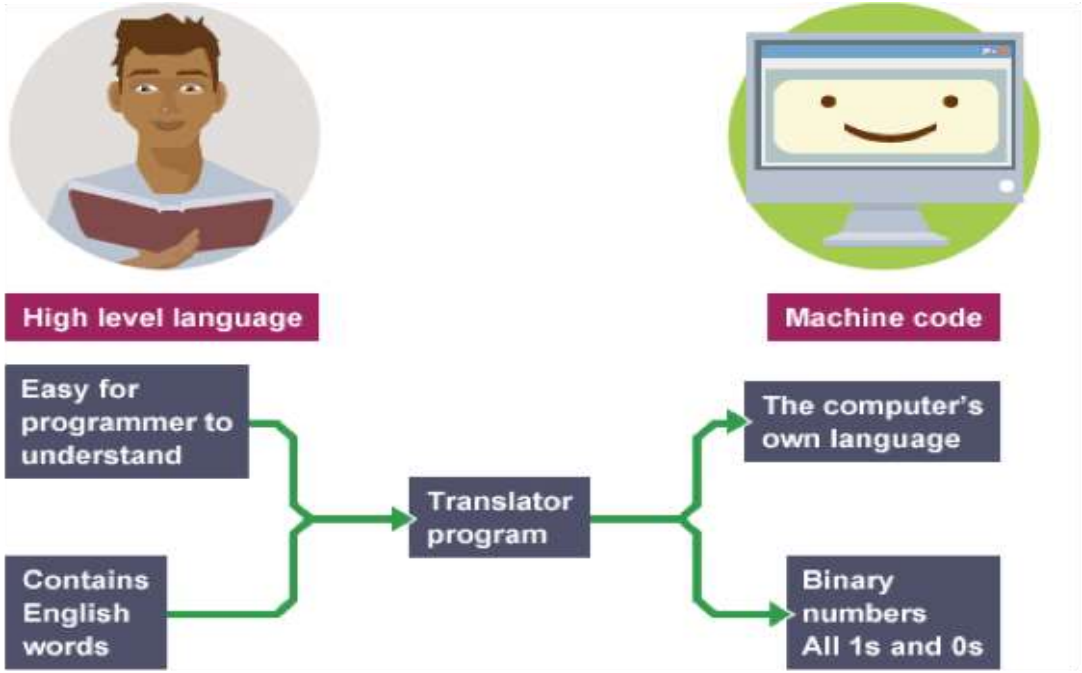
A solution was developed to have the translators generate to a kind of "half-way" standard intermediary code which could then be translated to each computers own specific machine code.

This half way language is called "**intermediate code**", often known as "**bytecode**". It is kind of useless on its own as it won't run without any further translation to turn it into machine code.

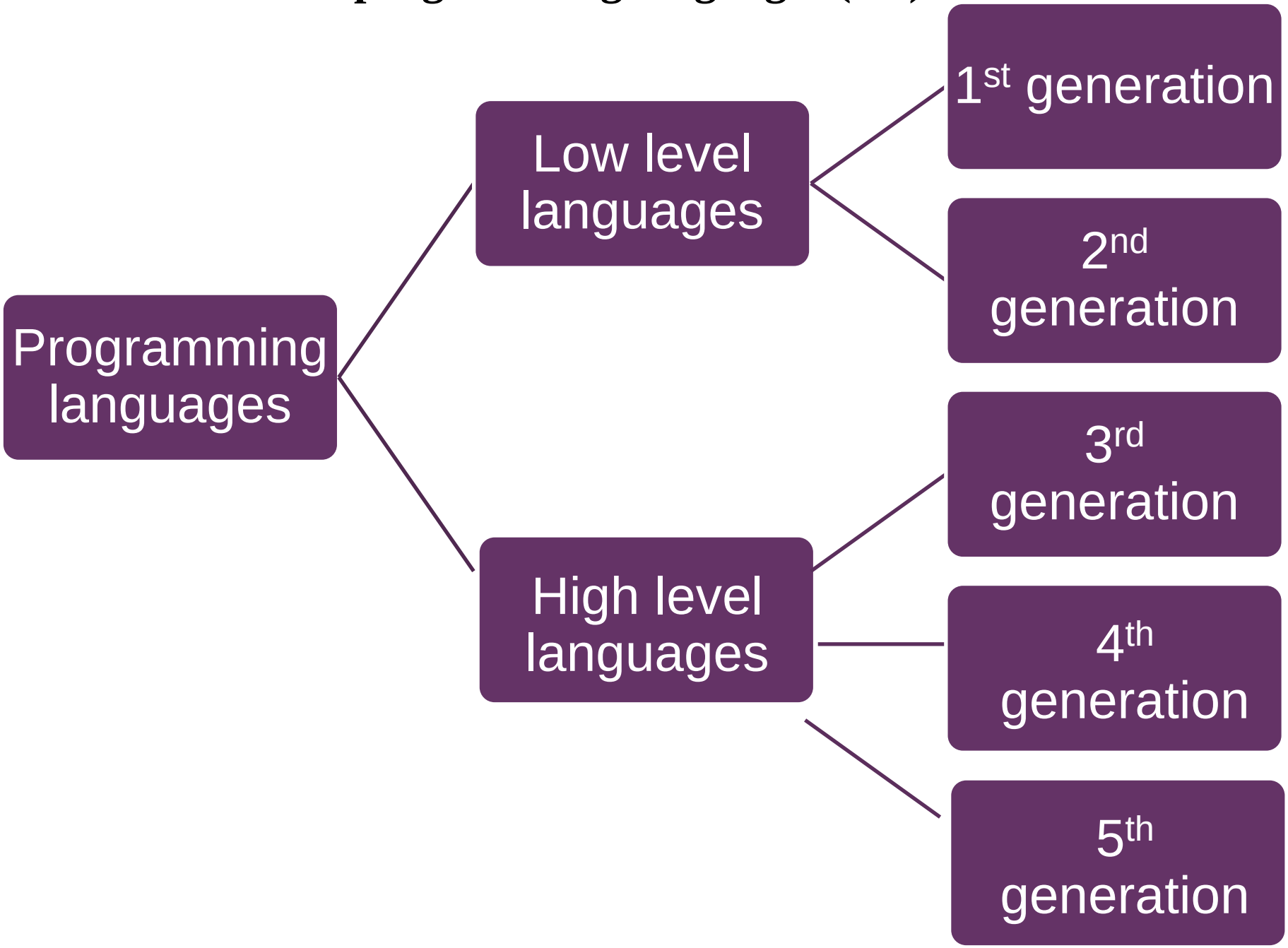
It does however run on a sort of 'pretend' machine that it was designed for, although this machine does not physically exist, it is installed on each make of computer, and it performs the job of taking the "generic" intermediate code and translating it into machine code specific for that machine.

This pretend machine is known as a "virtual machine".
Writing an interpreter to translate bytecode is a much easier task than writing an interpreter to translate high-level source code.
Bytecode is very portable and very compact.
Interpreting bytecode programs are faster than high-level source code programs.


Low level
1010010110111010
1001110110000111
0001110010110001
1011010110111010
0000111001010111
1001110010011101
Processing time
Slow
Fast
High level
sale_price = 1.66
if (sale_price > 2) {
discount = 0.1
}
else {
discount = 0.05
}
Processing time
Slow
Fast


High level language
Machine code
Easy for programmer to understand
Contains English words
Translator program
The computer's own language
Binary numbers All 1s and 0s

Classification of programming languages (PL)



Generations of Programming Languages

Generation	Language / Type
1	Machine language (Machine code)
2	Assembly language (Assembler)
3	Imperative languages (Basic, C, Python, Pascal, Java, Ruby, Fortran COBOL)
4	Logic languages (SQL,, HTML, CSS)
5	Prolog, Lisp, Mercury

Activity 3

Fill in the chart

- **Programming Languages**
- **Low level**
- **High level**
- **Imperative**
- **Declarative**
- **Procedural**
- **Assembler**
- **Machine code**
- **Object oriented**
- **Functional**
- **Logyc**

- **programming languages**
  - **LOW Level**
    - **Assembler , machine code**

  - **HIGH level**
    - **declarative**
      - **Functional**
      - **(LISP, Haskell)**

      - **Logyc**
      - **(Prolog, Mercury)**

    - **imperative**
      - **Procedural**
      - **(ADA, C, Paskal)**

      - **Object oriented (PHP, C++, JAvaScript, Pyhton)**

Activity 4

1. Place the following statements into the correct locations to show your understanding of the advantages of low-level languages and high-level languages.

Advantages / Disadvantages of high-level languages	Advantages/Advantages / Disadvantages of low-level languages
+	+
–	–

Easier to talk to hardware

Executes extremely f for embedded systems, real time systems, device drivers etc.

Occupies the least amo

Translation takes time

Often have access to man built-in library functions

Allows the programmer manipulate individual bits and bytes directly

he architecture dependent

Harder to modify and maintain

cker to program

tively easy to learn

More difficult to

Directly Interact with Hardware

Has to be translated

User friendly

Assessment criteria

Know that high-level languages include imperative high-level language

Understand the advantages and disadvantages of machine-code and assembly language programming compared with high-leve language programming

Explain the term ‘imperative high-level language’ and its relationship to low-level languages

Answer

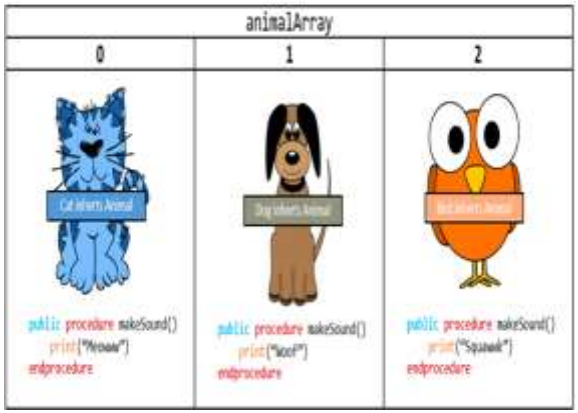
1. Place the following statements into the correct locations to show your understanding of the advantages of low-level languages and high-level languages.

Advantages / Disadvantages of high -level languages	Advantages / Disadvantages of low -level languages
Relatively easy to learn Easier to understand, maintain, improve, debug Often have access to many built-in library functions Machine architecture independent Easier and quicker to program User friendly	Executes extremely fast and efficiently Occupies the least amount of space possible Allows the programmer to manipulate individual bits and bytes directly Ideal for embedded systems, real time systems, device drivers etc. Directly Interact with the Hardware Easier to talk to hardware
Has to be translated Translation takes time	More difficult to learn Harder to modify and maintain

Compare different programming languages

```
1  predicates
2      parent(String,String)
3      male(String)
4      female(String)
5      brother(String,String)
6  clauses
7      parent("Tom","Jake").
8      parent("Janna","Jake").
9      parent("Tom","Tim").
10     parent("Janna","Tim").
11     male("Tom").
12     male("Tim").
13     male("Jake").
14     female("Janna").
15
16     brother(X,Y):-parent(Z,X),
```

```
run:
enter a side
5
enter b side
4
enter a side
8
2*a*b=40
2*b*c=64
2*a*b=40
area=184
СБОРКА УСПЕШНО ЗАВЕРШЕНА (общее время: 13 секунды)
```




```
package human;
import java.util.Scanner;

public class Human {
    public static void main(String[] args){
    Scanner sc = new Scanner(System.in);
    person person1 = new person();
    person1.name = sc.nextLine();
    person1.gender = sc.nextLine();
    person1.age = sc.nextByte();
    person1.student = sc.nextBoolean();

    System.out.println("\nPerson \n" +
            "name: " + person1.name +
            "\nGender: " + person1.gender +
            "\nAge: " + person1.age +
            "\nIs student: "+ person1.student);
    }
}
```

```
inp
sta 98
out
inp
sta 97
out
inp
sta 96
add 97
out
sub 98
out
hlt
```

Concise definitions!

Write a definition for the following three terms.

- Each definition must be 15 words or less.
- Each definition must make at least 3 valid points.

Interpreter	Definition here…
Compiler	Definition here…
Assembler	Definition here…

Interpreter	Takes one line of code, translates it, then runs it right away.
Compiler	Takes source code, translates it all into object code before allowing it to run.
Assembler	Translates a program written in assembly language into machine code.

Reflection

- What knows?
- What remained unclear
- What is necessary to work on