Session 1 28-08-2022

Tasks:

Task 1

What is OOD? why we use it? who use it?

	OOD
What IS	Object Oriented Design is a conceptual model for the architecture of object-oriented software. Software Engineer
Who Uses	Software Engineer
Why	design software that has a low cost of change

Task 2

What is the operating system core language?

OS	Language
Windows	kernel is developed mostly in C, with some parts in assembly language
Linux	Linux is also written mostly in C, with some parts in assembly.
MAC	Mac computers are also powered by C, since the OS X kernel is written mostly in C.
Mobile	iOS, Android and Windows Phone kernels are also written in C

Task 3

What are the first 10 programming languages appeared?

- 1843: Ada Lovelace's machine algorithm.
- 1944-45: Plankalkül.
- 1949: Assembly Language.
- 1949: Shortcode.
- 1952: Autocode.
- 1957: FORTRAN.
- 1958: ALGOL (Algorithmic Language)
- 1958: LISP (List Processor)
- 1959: COBOL (Common Business Oriented Language)
- 1964: BASIC (Beginner’s All-Purpose Symbolic Instruction Code)

Task 4

What are the type of languages that we could use in giving instructions?

Procedural languages
Procedural languages are based on the data viewing range of a code statement. Examples include Ada, BASIC, C/C++ and JavaScript.

Functional languages
Functional languages use stored data to perform recursive functions, which execute a process and then repeat it to solve any errors that arise during programming. Examples include Agda, Cuneiform, PureScript and APL.

Machine languages
Machine languages are made up of binary code, which is a series 0s and 1s that symbolize text or instructions for a computer program. One example of a machine language is Fortran.

Assembly languages
Assembly languages work in a similar way to machine languages by using short mnemonic codes to give the computer instructions. Examples include Lotus 1-2-3 and Turbo Pascal.

Logic programming languages
Logic programming languages add restrictions to statements made by developers that cause the computer to consider the possible outcomes of different actions. Examples include Prolog, ASP and Datalog.

Data-oriented languages
Data-oriented languages offer different ways to search and edit entity-relationship tables. Examples include Clarion, Gremlin, WebDNA and Wolfram Language.

Business-oriented languages
Companies use business-oriented languages to work with large quantities of data across a variety of different systems. Examples include SQL and COBOL.

Object-oriented languages
Object-oriented language identifies everything it encounters as objects that have internal and external data and then it performs based on moving these "objects" to where they need to be. Examples include Java, Visual Basic .NET, Ruby and Python.

Scripting languages
Scripting languages solve smaller programming issues and can be used to write operating system utilities. Examples include Perl, PHP, JavaScript and Python.

World Wide Web display languages
World Wide Web display languages are used to design web pages and provide them with the desired functions, such as page retrieval through links. Examples include HTML, XML and CGI.

Front end coding languages
Front end development languages are used to code the visual aspects of websites, games, software and apps. Examples include HTML, CSS and JavaScript.

Database programming languages
Database programming languages help to create databases and manipulate the way data is stored inside them. Examples include C++, COBOL, Java and Perl.

Rule-based languages
Rule-based languages implement rules once they are activated by certain conditions in a data set. Examples include AWK, CLIPS, Prolog and Wolfram Language.

Compiled languages
Compiled languages have been translated by computer programs from one programming language to another and convert information directly to code, which streamlines the programming process. Examples include ActionScript, Ballerina, C++ and ALGOL.

Back end coding languages
Back end coding languages code program servers so that web pages appear and function correctly. Examples include Python, Java and Ruby.

System languages
System languages can complete tasks like memory management or task management when programming an entire system. Examples include Swift, Rust, C++ and Nim.

Algorithmic languages
Algorithmic languages convey mathematical or symbolic computations and can use algebraic operations to convey information. Examples include Fortran, ALGOL, Lisp and C.

Command-line interface languages
Command-line interface languages use lines of text to send commands to computer programs. Examples include Batch, CLIST, TACL and 4DOS.

Task 5

What is the new programming language that have the same syntax of python and as fast as C?

The Peregrine programming language - A Python-like language that's as fast as C.

Task 6

List 1o programming languages open source and 10 not open source

Open Source	Not-Open Source
1.JavaScript	VBScript
2.Python	C#
3.PHP	MATLAB
4.Swift	COBOL
5.R Programming	IBM's mainframe
6.C++	AIX
7.Go	JAVA
8.Kotlin	
9.Scala	
10.Ruby	

Task 7

What are JavaScript advantages?

1. Speed Since JavaScript is an 'interpreted' language, it reduces the time required by other programming languages like Java for compilation. JavaScript is also a client-side script, speeding up the execution of the program as it saves the time required to connect to the server. 2. Simplicity JavaScript is easy to understand and learn. The structure is simple for the users as well as the developers. It is also very feasible to implement, saving developers a lot of money for developing dynamic content for the web. 3. Popularity Since all modern browsers support JavaScript, it is seen almost everywhere. All the famous companies use JavaScript as a tool including Google, Amazon, PayPal, etc. 4. Interoperability JavaScript works perfect with other programming languages and therefore numerous developers prefer it in developing many applications. We can embed it into any webpage or inside the script of another programming language. 5. Server Load	6. Rich Interfaces JavaScript provides various interfaces to developers for creating catchy webpages. Drag and drop components or sliders may give a rich interface to the webpages. This leads to improved user-interactivity on the webpage. 7. Extended Functionality Third-party add-ons like Greasemonkey (a Mozilla Firefox extension) allow the developers to add snippets of predefined code in their code to save time and money. These add-ons help the developers build JavaScript applications a lot faster and with much more ease than other programming languages. 8. Versatility JavaScript is now capable of front-end as well as back-end development. Back-end development uses NodeJS while many libraries help in front-end development like AngularJS, ReactJS, etc. 9. Less Overhead JavaScript improves the performance of websites and web applications by reducing the code length. The codes contain less overhead with the use of various built-in functions for loops, DOM access, etc.

As JavaScript operates on the client-side, data validation is possible on the browser itself rather than sending it off to the server. In case of any discrepancy, the whole website needs not to be reloaded. The browser updates only the selected segment of the page.	

Task 8

What is fragmentation?

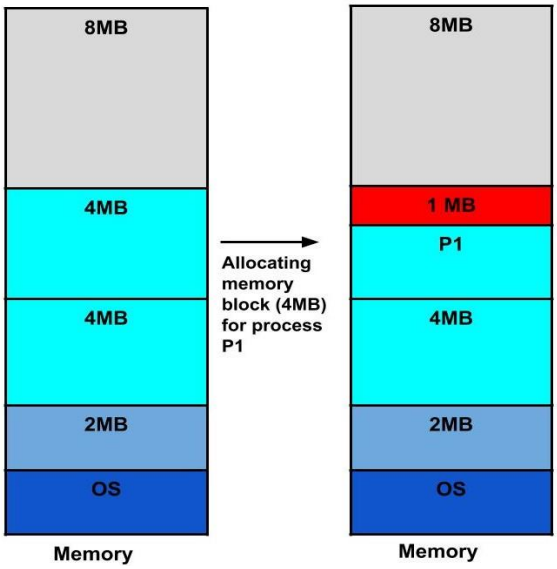
Fragmentation is an unwanted problem where the memory blocks cannot be allocated to the processes due to their small size and the blocks remain unused. It can also be understood as when the processes are loaded and removed from the memory they create free space or hole in the memory and these small blocks cannot be allocated to new upcoming processes and results in inefficient use of memory. Basically, there are two types of fragmentation:

- Internal Fragmentation
- External Fragmentation

Internal Fragmentation :
In this fragmentation, the process is allocated a memory block of size more than the size of that process. Due to this some part of the memory is left unused and this cause internal fragmentation.

Example: Suppose there is fixed partitioning (i.e. the memory blocks are of fixed sizes) is used for memory allocation in RAM. These sizes are 2MB, 4MB, 4MB, 8MB. Some part of this RAM is occupied by the Operating System (OS).

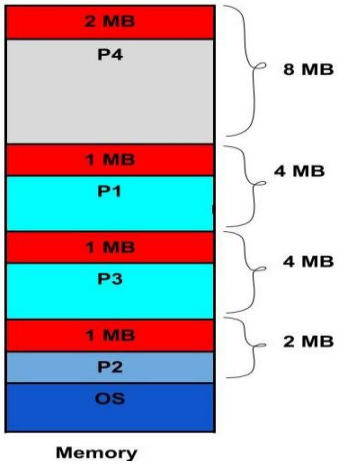
Now, suppose a process P1 of size 3MB comes and it gets memory block of size 4MB. So, the 1MB that is free in this block is wasted and this space can't be utilized for allocating memory to some other process. This is called internal fragmentation.



How to remove internal fragmentation?
This problem is occurring because we have fixed the sizes of the memory blocks. This problem can be removed if we use dynamic partitioning for allocating space to the process. In dynamic partitioning, the process is allocated only that much amount of space which is required by the process. So, there is no internal fragmentation.
External Fragmentation
In this fragmentation, although we have total space available that is needed by a process still we are not able to put that process in the memory because that space is not contiguous. This is called external fragmentation.

Example: Suppose in the above example, if three new processes P2, P3, and P4 come of sizes 2MB, 3MB, and 6MB respectively. Now, these processes get memory blocks of size 2MB, 4MB and 8MB respectively allocated.

So, now if we closely analyze this situation then process P3 (unused 1MB)and P4(unused 2MB) are again causing internal fragmentation. So, a total of 4MB (1MB (due to process P1) + 1MB (due to process P3) + 2MB (due to process P4)) is unused due to internal fragmentation.



Now, suppose a new process of 4 MB comes. Though we have a total space of 4MB still we can't allocate this memory to the process. This is called external fragmentation.

How to remove external fragmentation?
This problem is occurring because we are allocating memory continuously to the processes. So, if we remove this condition external fragmentation can be reduced. This is what done in paging & segmentation(non-contiguous memory allocation techniques) where memory is allocated non-contiguously to the processes. We will learn about paging and segmentation in the next blog.

Another way to remove external fragmentation is compaction. When dynamic partitioning is used for memory allocation then external fragmentation can be reduced by merging all the free memory together in one large block. This technique is also called defragmentation. This larger block of memory is then used for allocating space according to the needs of the new processes.