TOP-DOWN APPROACH

WORK PLAN

content

01 Introduction

Introduction

In programming, the top-down approach tackles complex problems by dividing them into more manageable subproblems.

Each subproblem is then solved individually, and these solutions are seamlessly integrated to achieve the overall solution. This technique, often called "Subproblem Reduction" among programmers

02 Benefits

Benefits

Well-known style

Helping programmers from similar backgrounds . Team leader incorporate familiar top-down methods to help them adjust quickly.

Efficient problem-solving

Help to tracing issues to their origin when they occur. Make it easier to find, diagnose, and resolve problems

Clearer overview

Top-down approach yields clear and organized code, minimizing confusion with centralized decisions and streamlined communication.

Faster execution

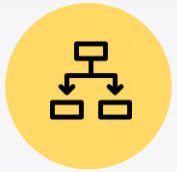
Quicker execution as decisions are made at a single level, enabling faster finalization, distribution, and implementation.

03 Step-in Top-down

Top-down approach in programming

Establishing the required steps before implementing a method provides clear insights and you can follow this:

SPECIFY THE PROBLEM: What are the inputs and outputs of the program?
What are the intended functions?



DIVIDE INTO SMALLER PROBLEMS: What are the primary tasks the program must execute?

Can we further break down these tasks into smaller sub-tasks?

CREATE INDIVIDUAL FUNCTIONS FOR EACH SUB-PROBLEM: This makes the program easier to change and use in different situations.

CALL THE SUB-FUNCTIONS FROM THE MAIN FUNCTION: The main function should just coordinate the running of the sub-functions.

TEST AND DEBUG THE PROGRAM: Ensure each part is solved correctly and check if the entire program works as intended.

04 Example

illustrated with c#

Below is a simplified version of the C# code that focuses only on displaying and adding products:

```
using System;
using System.Collections.Generic;

0 references
class Program
{
    0 references
    static void Main()
    {
        Console.WriteLine("Store Management");

        ProductManagement productManagement = new ProductManagement();

        productManagement.DisplayProducts();
        productManagement.AddProduct();

        Console.WriteLine("Press any key to exit.");
        Console.ReadKey();
    }
}
```

```
class ProductManagement
{
    private List<string> products = new List<string>();

    1 reference
    public void DisplayProducts()
    {
        Console.WriteLine("Product list:");
        Console.WriteLine(products.Count == 0
        ? "The list is empty." : string.Join(Environment.NewLine, products));
    }

    1 reference
    public void AddProduct()
    {
        Console.WriteLine("Enter product name:");
        products.Add(Console.ReadLine());
        Console.WriteLine("Product added successfully.");
    }
}
```

illustrated with C#

The program is a simple application that manages products in the store

Main Method: Still serves as the starting point, managing the main objects and functions of the program.

```
using System;
using System.Collections.Generic;

0 references
class Program
{
    0 references
    static void Main()
    {
        Console.WriteLine("Store Management");

        ProductManagement productManagement = new ProductManagement();

        productManagement.DisplayProducts();
        productManagement.AddProduct();

        Console.WriteLine("Press any key to exit.");
        Console.ReadKey();
    }
}
```

illustrated with C#

```
class ProductManagement
{
    private List<string> products = new List<string>();

    1 reference
    public void DisplayProducts()
    {
        Console.WriteLine("Product list:");
        Console.WriteLine(products.Count == 0
        ? "The list is empty." : string.Join(Environment.NewLine, products));
    }

    1 reference
    public void AddProduct()
    {
        Console.WriteLine("Enter product name:");
        products.Add(Console.ReadLine());
        Console.WriteLine("Product added successfully.");
    }
}
```

ProductManagement Class: This class is still responsible for managing product listings and related functions such as displaying and adding products.

DisplayProducts Method and AddProduct Method: Still smaller components, responsible for specific functions.

05 Real-world

What Tech Companies Use the Top-Down Approach?

Here are some examples of tech companies that are known for incorporating the top-down approach in their software development processes:

1.Microsoft

2.IBM

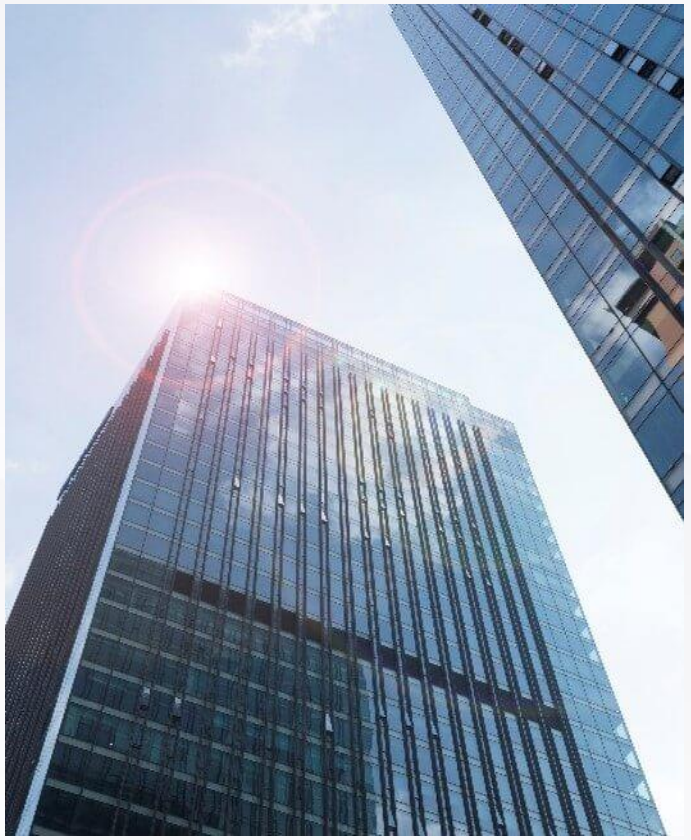
3.Oracle

4.Google

5.Amazon

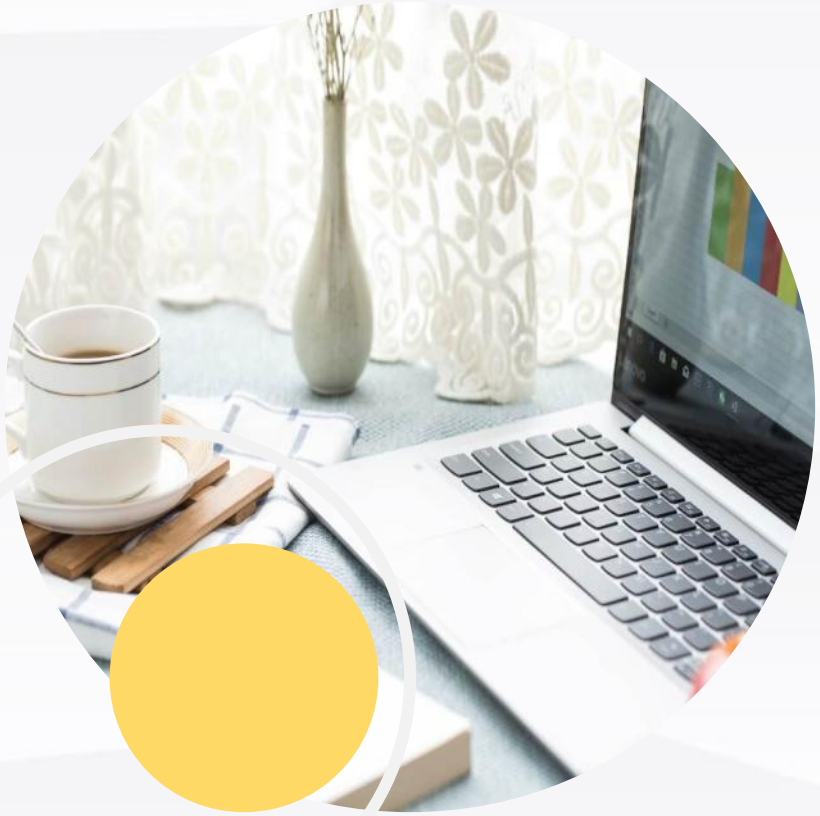
6.Facebook

7.Apple



06 Conclusion

Conclusion



Overall, the top-down approach promotes a logical and organized development process, leading to software solutions that are not only scalable and maintainable but also aligned with the overall goals of the project.

Its adaptability makes it a valuable methodology in various tech companies for creating robust and efficient software systems.

Thanks for watching

WORK PLAN