Sample Questions for TechM .NET Track

MCQ:

Topic: OOPS

DESCRIPTION

Which of the following is the most crucial benefit of inheritance?

Option 1: **(Correct Answer)**

Reusability

Option 2:
Better organization of code

Option 3:
Make application code more flexible to change

Option 4:
Data hiding

Option 5:
Minimize the amount of duplicate code in an application

Topic: C# 5 .0 Features

DESCRIPTION

Eva is working with c# 5.0. She wants to set information about the caller source code file and the caller member's name. Which of the following functions she should use to achieve the same?

Option 1: **(Correct Answer)**

CallerFile
CallerMemberName

Option 2:
CallerFilePath
CallerMemberName

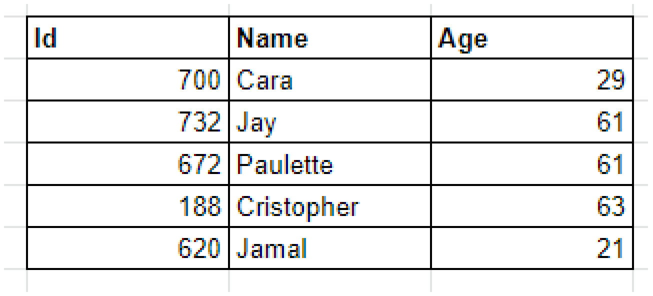
Option 3:
CallerFile
CallerMember

Option 4:
None of the Above

Topic: DML

DESCRIPTION

For the data below, write a query to get all the users whose name starts with the letter '**C**'.



Id	Name	Age
700	Cara	29
732	Jay	61
672	Paulette	61
188	Cristopher	63
620	Jamal	21

Option 1: SELECT * FROM users WHERE name LIKE 'C%'; **(Correct Answer)**

Option 2: SELECT * FROM users WHERE name LIKE 'C_';

Option 3: SELECT * FROM users WHERE name = 'C%';

Option 4: SELECT * FROM users WHERE name LIKE '%C';

Topic: ASP.NET MVC 5(Controllers)

DESCRIPTION

Which of the following is the **correct** way to define a **constructor** that accepts a parameter in an ASP.NET MVC controller?

Option 1: public MyController()
Option 2: public MyController(string parameter)
Option 3: public void MyController(string parameter)
Option 4: public MyController(parameter) **(Correct Answer)**

Topic: Azure(Branching)

DESCRIPTION

Which of the following is a recommended best practice for using pull requests in Azure DevOps?

Option 1: Merge pull requests as soon as possible to avoid conflicts with other changes.

Option 2: Do not use pull requests for small changes, as they are unnecessary.

Option 3: Assign reviewers to the pull request and incorporate their feedback before merging the changes into the main branch.**(Correct Answer)**

Option 4: Use pull requests only for changes that have already been extensively tested and validated.

Coding:

DESCRIPTION

Employee Operation:

This problem is related to handling exceptions when the objects are not initialized.

Your task here is to implement a **C#** code based on the following specifications. Note that your code should match the specifications in a precise manner. Consider **default visibility** of classes, data fields and methods are public unless mentioned otherwise.

Specifications:

```
class definition:
        class Employee:
                visibility : default
                member property : Id
                return type : int
                visibility : public
                member property : Name
                return type : string
                visibility : public
                member property : Department
                return type : string
                visibility : public
                member property : CTC
                return type : double
                visibility : public
        class Operation:
                visibility : default
                member variable : EmployeeList
                return type : IList<Employee>
                visibility : public
                member definition : AddEmployee(Employee employee)
                return type : void
                visibility : public
                member definition : GetHighestPaidEmployee()
                return type : Employee
                visibility : public
```

Task:

class Employee:

- Define the class as mentioned in specification.

class Operation:

- **AddEmployee(Employee employee):** Accepts an employee and it should be added to EmployeeList. If the EmployeeList is not initialized, then NullReferenceException should be caught and EmployeeList should be initialized and employee should be added to the list.
- **GetHighestPaidEmployee():** If the EmployeeList is null or empty then it should throw InvalidOperationException with the message **"Cannot get the highest paid employee"**, otherwise it should return the employee which is having the highest CTC.

Sample input:

```
var operation = new Operation();
   operation.AddEmployee(new Employee
   {
    Id = 25,
    Name = "John",
    CTC = 560000,
    Department = "R&D"
   });
   operation.AddEmployee(new Employee
   {
    Id = 48,
    Name = "Michal",
    CTC = 860000,
    Department = "Operations"
   });
   operation.AddEmployee(new Employee
   {
    Id = 10,
    Name = "Kerry",
    CTC = 600000,
    Department = "Finance"
   });
   var highestPaidEmployee = operation.GetHighestPaidEmployee();
   Console.WriteLine($"Id: {highestPaidEmployee.Id}, Name: {highestPaidEmployee.Name},
Department: {highestPaidEmployee.Department}, CTC: {highestPaidEmployee.CTC}");
```

Sample output:

```
Id: 48, Name: Michal, Department: Operations, CTC: 860000
```

Note:

- You can make suitable function calls and use **the RUN CODE** button to check your **main()** method output.

Debugging Coding:

DESCRIPTION

Problem Statement

You are given a **C#** program that generates and prints the Fibonacci sequence up to a given number **n**. However, there is a bug in the code that needs to be identified and fixed.

The Fibonacci function is responsible for calculating the Fibonacci numbers, but a subtle mistake has been introduced. Your task is to debug the code and correct the error in the Fibonacci function so that it produces the correct Fibonacci sequence.

Instructions:

1. Ensure the program prints the correct Fibonacci sequence up to the input **n**.
2. Test the program with different values of **n** to verify its correctness.

Ensure to maintain the Fibonacci function's recursive nature while fixing the bug.

Sample Input

```
5
```

Sample Output

```
0 1 1 2 3
```

TASK:

- A code stub containing buggy code will be loaded by default.
- You have to debug the solution code so that all test cases pass.
- You can click on the **Run Code** button to check for compilation errors.
- You can click on the **Verify** button to check for run-time errors (against **sample test cases**).
- You can click on the **Submit** button to check for run-time errors (against **actual** test cases).

Full Satck

DESCRIPTION

Problem Statement

Implement a RESTful API for managing projects in a project management system. The system should support the retrieval of a list of all projects and the details of a specific project by its Id.

The Project class represents a project entity with the below properties:

- **ProjectId**: the unique Id of the project (int)
- **ProjectName**: the name of the project (string)
- **TechnologyUsed**: the technology used in the project (string)

Here is an example of a Project JSON object:

```
[
  {
    "projectId": 1,
    "projectName": "Library API Service",
    "technologyUsed": "Django & React"
  },
  {
    "projectId": 2,
    "projectName": "Mobile API Service",
    "technologyUsed": "Flutter & Angular"
  }
]
```

You are provided with the implementation of the models required for all the APIs. Your task is to implement a set of REST services that exposes the endpoints, allows for listing all the projects, and also retrieves the details of a specific project based on the Id.

Task-1: **GET request to /api/Project**

1. Retrieve a list of projects and return them as JSON.
2. The response code is 200, and the response body is a JSON object with the message "**List of projects**" and the details of the project.
3. If there are no such projects, return status code 404, and the response body will be a simple string message "**Projects Not Found**".

Task-2: GET request to /api/Project/{id}

1. Retrieve a single project by its Id.
2. The response code is 200, and the response body is a JSON object with the message "**Id Found**" and the details of the project.
3. If there is no such project for the given Id, return status code 404. The response body will be a simple string message "**Given Id Not Found".**

Complete the given project so that it passes all the test cases when running the provided unit tests. Files in which you need to write the code:

1.) ProjectApp.WebAPI/Controllers/ProjectController.cs

- Implement the Get Methods here

2.) ProjectApp.WebAPI/Services/ProjectServices.cs

- Retrieve all projects from the database asynchronously using ToListAsync().
- Retrieve a project by its Id from the database asynchronously using FindAsync(Id).

<u>Example Requests and Responses:</u>

GET request to /api/Project:

Example Response Body (HTTP 200 OK):

```
{
  "msg": "List of projects",
  "projects": [
    {
      "projectId": 1,
      "projectName": "Library API Service",
      "technologyUsed": "Django & React"
    },
    {
      "projectId": 2,
      "projectName": "Mobile API Service",
      "technologyUsed": "Flutter & Angular"
    }
  ]
}
```

GET request to api/Project/{id}:

Example Response Body (HTTP 200 OK):

```
{
  "msg": "Id Found",
  "project": {
    "projectId": 1,
    "projectName": "Library API Service",
    "technologyUsed": "Django & React"
  }
}
```

Guidelines to connect to SQL Server database

Note:

The backend of the application uses a **SQL Server** database.

Please use the following commands from the terminal in the IDE to access it,

To access the database:

```
/home/user/workspace# mssql-cli -S localhost -U SA -P 'P@ssw0rd'
```

Once the application is built, it will create the **projectDB** database and the Project table. To access the tables, run the following command.

```
master> use projectDB;

projectDB;> select name from sys.tables;
Time: 0.452s
+---------+
| name    |
|---------|
| Project |
+---------+
(1 row affected)
projectDB;>
```

You can use the **Sqlite** shell to view the data and query the tables.