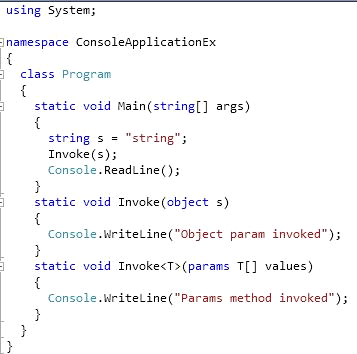
Dot Net Part 1

Answer minimum 3 out of the following questions.

1. You are modifying an ASP.NET MVC web application for a client who requires that the application must be viewable on an Android device in a UI that is native to the device. Describe how we can achieve this.

**Answer**: **Responsive Design using Bootstrap libraries and Mobile Views** in MVC. **The Viewport Meta Tag** - Most mobile browsers define a virtual browser window width (the viewport) that's much larger than the actual width of the mobile device. This allows mobile browsers to fit the entire web page inside the virtual display. Users can then zoom in on interesting content. **CSS Media Queries** - @media only screen and (max-width: 850px). They allow you to create rules that override the default CSS rules for specific browsers.

1. Below is the sample what will be the output. Provide brief explanation.
   1. 

**Answer**: Output will be “**Params method invoked”.** We are not sure of the number of arguments passed as a parameter. Complier will prefer to call method with generic parameter and will not call method were complier require to boxing the input parameter. The first method will call, if instead of Object parameter as a string.

1. Write a sample code to force the Dispose method to be called automatically.

**Answer**:

public class Resource : IDisposable

{

private IntPtr nativeResource = Marshal.AllocHGlobal(100);

private AnotherResource managedResource = new AnotherResource();

public void Dispose()

{

Dispose(true);

GC.SuppressFinalize(this);

}

~Resource()

{

Dispose(false);

}

protected virtual void Dispose(bool disposing)

{

if (disposing)

{

if (managedResource != null)

{

managedResource.Dispose();

managedResource = null;

}

}

if (nativeResource != IntPtr.Zero)

{

Marshal.FreeHGlobal(nativeResource);

nativeResource = IntPtr.Zero;

}

}

}

* 1. Define what is an Interface and why do we use it? Write a sample code to show/print data in many pages of an application using an abstract method.

**Answer**: Using **IDisposable** Interface, we can define. This is because the garbage collector is very efficient at reclaiming unused managed objects, but it is unable to reclaim unmanaged objects.

* 1. Which members have the capability to override in the child class? Write a sample code to override the members of the Base class.

Answer:

public class StudentName

{ }

public abstract class Course

{

public abstract void AddStudent(StudentName sn, int f);

public abstract decimal CalculateIncome();

}

public abstract class WritingCourse : Course

{

override public void AddStudent(StudentName sn, int f)

{

//Add student

}

override public abstract decimal CalculateIncome();

}

public class BusinessWritCourse : WritingCourse

{

override public void AddStudent(StudentName sn, int f)

{ base.AddStudent(sn, 0); }

override public decimal CalculateIncome()

{

return (decimal)3.4;

}

}

public class SewingCourse : Course

{

public override void AddStudent(StudentName sn, int f)

{

}

public override decimal CalculateIncome()

{

return (decimal)10;

}

}

1. You created a Microsoft ASP.NET application by using the Microsoft .NET Framework version 3.5. All the content pages in the application use a single master. The master page uses a static navigation menu to browse the site. You need to ensure that the content pages can optionally replace the static navigation menu with their own menu controls. What will you do?

Dot Net Part 2

Answer minimum 2 out of the following questions.

* Write Code to create hash and salt complex passwords that contains random characters with below conditions. Also provide 5 sample passwords that are generated.
* Password length should contain minimum 6 characters
* It should contain Numbers
* It should contain upper and lower cases
* It should contain special symbols

Answer:

class Program

{

static void Main(string[] args)

{

bool includeLowercase = true;

bool includeUppercase = true;

bool includeNumeric = true;

bool includeSpecial = true;

bool includeSpaces = false;

int lengthOfPassword = 10;

for (int i=0; i < 5; i++){

string password = GeneratePassword(includeLowercase, includeUppercase, includeNumeric, includeSpecial, includeSpaces, lengthOfPassword);

while (!PasswordIsValid(includeLowercase, includeUppercase, includeNumeric, includeSpecial, includeSpaces, password))

{

password = GeneratePassword(includeLowercase, includeUppercase, includeNumeric, includeSpecial, includeSpaces, lengthOfPassword);

}

Console.WriteLine(password);

System.Threading.Thread.Sleep(100);

}

Console.ReadLine();

}

public static string GeneratePassword(bool includeLowercase, bool includeUppercase, bool includeNumeric, bool includeSpecial, bool includeSpaces, int lengthOfPassword)

{

const int MAXIMUM\_IDENTICAL\_CONSECUTIVE\_CHARS = 2;

const string LOWERCASE\_CHARACTERS = "abcdefghijklmnopqrstuvwxyz";

const string UPPERCASE\_CHARACTERS = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

const string NUMERIC\_CHARACTERS = "0123456789";

const string SPECIAL\_CHARACTERS = @"!#$%&\*@\";

const string SPACE\_CHARACTER = " ";

const int PASSWORD\_LENGTH\_MIN = 6;

const int PASSWORD\_LENGTH\_MAX = 128;

if (lengthOfPassword < PASSWORD\_LENGTH\_MIN || lengthOfPassword > PASSWORD\_LENGTH\_MAX)

{

return "Password length must be between 6 and 128.";

}

string characterSet = "";

if (includeLowercase)

{

characterSet += LOWERCASE\_CHARACTERS;

}

if (includeUppercase)

{

characterSet += UPPERCASE\_CHARACTERS;

}

if (includeNumeric)

{

characterSet += NUMERIC\_CHARACTERS;

}

if (includeSpecial)

{

characterSet += SPECIAL\_CHARACTERS;

}

if (includeSpaces)

{

characterSet += SPACE\_CHARACTER;

}

char[] password = new char[lengthOfPassword];

int characterSetLength = characterSet.Length;

System.Random random = new System.Random();

for (int characterPosition = 0; characterPosition < lengthOfPassword; characterPosition++)

{

password[characterPosition] = characterSet[random.Next(characterSetLength - 1)];

bool moreThanTwoIdenticalInARow =

characterPosition > MAXIMUM\_IDENTICAL\_CONSECUTIVE\_CHARS

&& password[characterPosition] == password[characterPosition - 1]

&& password[characterPosition - 1] == password[characterPosition - 2];

if (moreThanTwoIdenticalInARow)

{

characterPosition--;

}

}

return string.Join(null, password);

}

public static bool PasswordIsValid(bool includeLowercase, bool includeUppercase, bool includeNumeric, bool includeSpecial, bool includeSpaces, string password)

{

const string REGEX\_LOWERCASE = @"[a-z]";

const string REGEX\_UPPERCASE = @"[A-Z]";

const string REGEX\_NUMERIC = @"[\d]";

const string REGEX\_SPECIAL = @"([!#$%&\*@\\])+";

const string REGEX\_SPACE = @"([ ])+";

bool lowerCaseIsValid = !includeLowercase || (includeLowercase && Regex.IsMatch(password, REGEX\_LOWERCASE));

bool upperCaseIsValid = !includeUppercase || (includeUppercase && Regex.IsMatch(password, REGEX\_UPPERCASE));

bool numericIsValid = !includeNumeric || (includeNumeric && Regex.IsMatch(password, REGEX\_NUMERIC));

bool symbolsAreValid = !includeSpecial || (includeSpecial && Regex.IsMatch(password, REGEX\_SPECIAL));

bool spacesAreValid = !includeSpaces || (includeSpaces && Regex.IsMatch(password, REGEX\_SPACE));

return lowerCaseIsValid && upperCaseIsValid && numericIsValid && symbolsAreValid && spacesAreValid;

}

* Create web API for storing cards data (card number, card state, card issuer date, card expired date)

**Answer**:

public class CCardDetails

{

public int CardNumber { get; set; }

public string CardState { get; set; }

public DateTime CardIssueDate { get; set; }

public DateTime CardExpiryDate { get; set; }

}

public class CCardDetailsController : ApiController

{

private MVCWebAPIContext db = new MVCWebAPIContext();

[HttpPost]

public IHttpActionResult PostCCardDetails([FromBody] CCardDetails ccarddetails)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

db.CCardDetails.Add(ccarddetails);

db.SaveChanges();

return CreatedAtRoute("DefaultApi", new { id = ccarddetails.AutoId }, ccarddetails);

}

}

* Create small application for doing: -
* Add new card in data base
* Search about any cards or all cards
* Update the state of any card
* Delete any card
* Use classes to Serialization and Deserialization

Notes: -

* Card number must be unique and generated it randomly.
* The length of card number must be 14 digits.
* Write a code to get all prime numbers between two numbers with condition this number should be prime number even if we reverse it like (179 and 971)

**Answer**:

class Program

{

static void Main(string[] args)

{

//

Console.Write("Enter a Range : ");

int num1 = Convert.ToInt32(Console.ReadLine());

int num2 = Convert.ToInt32(Console.ReadLine());

Console.Write("Prime number between {0}-{1} - ",num1,num2);

for (int i = num1; i < num2; i++) {

string j;

if (IsPrime(i) && IsPrime(Convert.ToInt32(Reverse(i.ToString(),out j)))) {

Console.Write("{0}-{1}, ", i,j);

}

}

Console.ReadLine();

}

static string Reverse(string text,out string t)

{

t = "";

if (text == null) return null;

char[] array = text.ToCharArray();

Array.Reverse(array);

t = new String(array);

return t;

}

static bool IsPrime(int s)

{

bool isPrime = true;

int temp = 0;

for (int i = 2; i <= s / 2; i++){

temp = s % i;

if (temp == 0){

isPrime = false;

break;

}

}

return isPrime;

}

}