Terms/Keywords

(Not everything is filled in/there are definitely some misspelled stuff)

C# & .NET

ANATOMY OF PROGRAM .cs

using.System- refers to one of many libraries of the .NET Framework

Namespace - a way to organize code and classes and prevent naming conflicts which allows two classes to have the same name to exist in different namespaces in the same project. Namespace are used to organize the classes. It helps to control the scope of methods and classes in larger . Net programming projects. The biggest advantage of using namespace is that the class names which are declared in one namespace will not clash with the same class names declared in another namespace.

keyword: namespace <EnterNamespaceName>

Static - means that it belongs to Program class and not an object of the Program
Void - means that this method odes not have a return value.
Main - Method main is the starting point to write your program.

```
{
    class <name> -class keyword is to define and create objects (almost everything in C# is
    an object.
    {
    <field/variables> - hold data value
    int <number>
    string <name>
    static void Main()
        {
            CODE statements should end with ;
        }
    }
}
```

C# - An Object-Oriented programming language. It's a general-purpose meaning there are many applications you can build using C# for example console applications Desktop applications, web services and applications and much more. It is a multi-paradigm programming language[still trying to figure that part out]. The created of C# took the best parts of java, c, c++ and innovated by introducing new concepts like types, properties and events. Another benefit of C# is garbage collector, So for example users don't need to worry about memory management. C# is a strongly type language, every variable and constant has a type as does every expression that evaluates to a value.

.NET - provides an environment for both developing, and executing, multi-platform applications. The framework has been designed in such a way that it can be used from several languages

like C++, C#, Visual Basic, Jscript etc. All these languages can access the framework as well as communicate with each other.

SOLID Principles - golden rules used by object-oriented developers since the early 2000s. They set the standard of how to program in OOP languages and now beyond into agile development and more. SOLID programs scale better, cost less time to work with, and can more easily respond to change. Employers will always prefer a candidate with a strong grasp of SOLID principles.

1. *Single Responsibility -* Objects should have one and only one purpose.
2. *Open-Close -* Open for extension, closed for modification.
3. *Liskov Substitution Principle -* Derived objects should be substitutable for their base implementations.
4. *Interface Segregation* - Users of interfaces should not be dependent upon methods that do not use.
5. *Dependency Inversion-* Concrete implementations should be based upon abstract definitions.

Object-Oriented-Programming Language 4 Pillars -

1. *Abstraction* - A way to simplify
2. *Polymorphism* -
3. *Inheritance* - Create an IS-A relationship.
4. *Encapsulation* - a way to protect data

INTERFACE-Interface can be used to enforced the presence of a method in any class that implements. It helps us achieve multiple inheritance.

Keyword: interface.

Syntax - in computer programming as the concept of giving specific word sets in specific orders to computers so that they do what we want them to do

Repository - A folder in the internet(GitHub) used to store code. Depending if it is public or private, others can view, edit, pull, push. To make sure you are in the repo change directory to where you clone repo. You should see (main).
Ex.<CurrentUser>/<Machine>MINW64~/Directory/repo name(main) In your repo you can add, edit, remove and push.

Binary-

Binary	Decimal
0000	0
0001	1
0010	2

0011	3
1111	15

CONDITIONALS
if ->Comparison
else-if -> Testing if a second comparison is true.
else -> if all else fails do (condition)\
== -is equal to
> or < Greater or less than.
>= or <= Greater/equal or less/equal.
!= -Not equal to

|| -> Logical OR between two things one needs to be true
1 = True/Yes/On; 0 = False/No/Off
0 || 0 = 0
1 || 0 = 1
0 || 1 = 1
1 || 1 = 1
&& -> Logical AND between two things one needs to be true
1 = True/Yes/On; 0 = False/No/Off
0 && 0 = 0
1 && 0 = 0
0 && 1 = 0
1 && 1 = 1
Bash shell - Converts code to CPU to understand.

DATA TYPES

1. *Primitive data types* -the main built-in types and could be used to build other data types.
    1. int- All whole numbers.
    2. char- Unicode character.
    3. short- Whole numbers.
    4. float- Real numbers
    5. bool- Logical True/False.
    6. byte- Whole numbers.
    7. double- All real numbers.
    8. long- Whole numbers (HUGE to tiny)

Type Conversions - Converting one type of date to another type know as Type Casting.
There are two types

1. Implicit type- preform by C# in type safe manner. These conversions are performed by C# in a type-safe manner. For example, are conversions from smaller to larger integral types and conversions from derived classes to base classes.

2. Explicit type- done by users using predefined functions Explicit conversions require a cast operator. These conversions are done explicitly by users using the pre-defined functions. Explicit conversions require a cast operator.
1. *Parsing*- Analyze (a string or text) into logical syntactic components, typically in order to test conformability to logical grammar. Converts string that represents a .NET base type into that base type.
2. *Casting* -> A value is converted from one data type to another. Casting does not change the variables value, the value remains the same type.
    Keyword: as - To upcast.
    Keyword: is - To downcast.

Log Messages

1. *Debug*- Useful development tool to confirm or verify the status or operation of your code.
2. *Info*- Important info. about the regular operation of a program or service.
3. *Tracing*- Useful info. about actions taken leading up to an error or fatal error.
4. *Warnings*- Notifies user of some oddity-version depreciated line ending.
5. *Errors* - anything fatal to an operation, but not to the entire application
6. *Fatal/Critical* - any error that is forcing the full shutdown of an application/service
7. *Exceptions* - a problem that arises during the execution of a program, that provides. A way to transfer control from one part of a program to another. Exceptions in C# provide a structured, uniform, and type-safe way of handling both system level and application level error conditions. The exception mechanism in C# is quite similar to that of C++, with a few important differences:

In C#, all exceptions must be represented by an instance of a class type derived from System.Exception.
In C++, any value of any type can be used to represent an exception.
In C#, a finally block (The try statement) can be used to write termination code that executes in both normal execution and exceptional conditions. Such code is difficult to write in C++ without duplicating code.
In C#, system-level exceptions such as overflow, divide-by-zero, and null dereferences have well defined exception classes and are on a par with application-level error conditions.
An exception is similar to an if else, because they are working through conditionals.
    Keywords: try, catch, finally, throw.

Loops

1. *For*-Max number of loops, set number of iterations, must use numerical comparison. If the number of iteration is fixed it is recommended to use for loop vs do-while.
2. *Do-While* - Are used to iterate a part of the program several times. If the number of iterations is not fixed and you must have to execute the loop at least once it is recommended to use do-while.
3. *While* - use a single test condition. If it's true it will run again. There are used to iterate a part of the program several times. If the number of iteration is not fixed while is recommend vs the for loop.

Collections

1. *Generic*-Performs faster and minimize exceptions by giving compile time errors. Hold elements of the same datatype.
    1. *List<T>*
    2. *Dictionary<Tkey, Tvalue>* -Represents a collection of keys and values.
    3. *Queue<T>*
    4. *Stack<T>*
    5. *Hashset<T>* -Represents a set of values.
2. *Non-Generic* -Hold elements of different datatypes.
    1. *ArrayLsit*
    2. *SortedList*
    3. *Stack*
    4. *Queue*
    5. *Hashtable*
    6. *Bitarray*

Array-Structure representing a fixed length order collection of values or objects with the same type.
List - List of objects which can be accessed by index. It supports storing values of specific type without casting to or from an object. More flexible than an array.
Queue -First-in, first-out collection of objects. Need to use System.Collections.Generic Keywords: Queue<T>, Enqueue()->adds, *Dequeue() or Peek()->retrieve elements.
Stack - Last in, first out. Keyword: stack<T>, push()->add, pop() or peek()-> retrieve. Also come from System.Collections.Generic

Method -A block of code which only runs when it is called. You can pass data (parameters) into a method. They are used to preform certain actions (functions). To create a method define the method name and () like for example the main method is written as such: Main()

1. *Overloading*- creating multiple versions of a method or constructor that accept different parameters(#of parameters, types, ets) and accomplish the same task. share the same name, but different parameters.
2. *OVERRIDING* - Method overriding is a technique that allows the invoking of function from another class in the derived class. Creating a method in the derived class with the same
3. dsaf

PARAMETERS (REWRITE!!*)
•Value - Creates a copy of the parameter passed so modification does not affect each other.
•Reference - Keyword: ref -causes a method to refer to the same variable that was passed into the method. Making changes to original.
•Out- keyword: out -u sed when you want to return more than one value.
•Array - keyword: parms - must be 1 dimensional array Allows us to send a variable number of parameters to a function

Visual Studio 2022
ASPE.NET and web development

Field-
Property-

public string Number {get} - doesn't change

in .NET:
there are multiple languages
C#, F#, VB.NET
(C++/CLI)
Multiple Runtimes:
Window, MAC, Linx, 32-bit, 64-bit
but also mobile, embedded systems/IoT

different runtime with different priorities(.NET6, mono)

Multiple Frameworks

.NET Framework
standardizes as: Common Language Infrastructure (CLI)
SOURCE CODE -->
COMPILER-->
ASSEMBLY-->
RUNTIME:
Virtual Execution System(VES)
Just-in-time complier (JIT)
Executes that native code
Implement memory management(garbage collection)

Different .NET implementations:
-.NET Framework(VES=Common Language Runtime(CLR)) [windows only]
-Mono was reversed engineer to run on linux [linux/mac only]
-.NET Core v1-3.1
-.NET v5-6 next version of .NET Core, rebranded

the common denominator of .NET Framework and .NET Core is called .NET Standard.

Serialization- Take in data structure and turn it into a string and be able to send. To serialize something is just to turn it into a blob of either binary or text. To desterilize is to take that blob of binary or text and turn it back into something like an object instance like a list of names can be

serialized into XML and make it easier to send over the internet, save to file, or deserialize back to object later.

Testing - Why? Real world we have to be able to prove what we created works. Testing is a way to prove it does the job. In testing add reference by right clicking on dependencies. Why isn't debugging/build good enough? Because we could get no errors but what we want the code to do could be wrong. Example We write a calculator and it will output a number but not the right number (PEMDAS). 1+2*(4-1) should be 7 but since we build and got no errors our output could be wrong like saying its 11. Testing helps us make sure its doing what we want it to do, like in this case know to use PEMDAS.
Unit Testing- The triple A's Arrange, Act, Assert.

1. Act
2. Arrange
3. Assert

****DEBUDDING****
We can add break points
Debug.WriteLine(); -works in debug mode

Type System

The common type system
It's important to understand two fundamental points about the type system in .NET:

It supports the principle of inheritance. Types can derive from other types, called base types. The derived type inherits (with some restrictions) the methods, properties, and other members of the base type. The base type can in turn derive from some other type, in which case the derived type inherits the members of both base types in its inheritance hierarchy. All types, including built-in numeric types such as System.Int32 (C# keyword: int), derive ultimately from a single base type, which is System.Object (C# keyword: object). This unified type hierarchy is called the Common Type System (CTS). For more information about inheritance in C#, see Inheritance.
Each type in the CTS is defined as either a value type or a reference type. These types include all custom types in the .NET class library and also your own user-defined types. Types that you define by using the struct keyword are value types; all the built-in numeric types are structs. Types that you define by using the class or record keyword are reference types. Reference types and value types have different compile-time rules, and different run-time behavior.

1. *Namespace*
2. *Classes*
3. *Records*
4. *Interfaces*
5. *Generics*
6. *Anonymous Types*

Stack- for local variables

Heap-pretty much everything else. Objects and Reference types are stored on the Heap.

****STRUCT****
struct->keyword Example (<public, private, protect> struct <name>)
user define types.

- *Modifiers*

1. *readonly*- Keyword: readonly. Can be used in four contexts:
    1. In a field declaration it indicates that assignment to the field can only occur as part of the declaration or in constructor in the same class. A readonly field can be assigned and reassigned multiple times within the field declaration and constructor.
    2. readonly Can not be assigned after the constructor exits. This rule has two different implications.
        1. VALUE TYPES- because value types directly contain their data, a field that is a readonly value type is immutable (can not be changed).
        2. REFERENCE TYPES - contains a reference to their data, a field that is a readonly reference type must always refer to the same object. That object is not immutable. The readonly modifier prevents the field from being replaced by a different instance of the reference type. However the modifier doesn't prevent the instance data of the field from being modified through the read-only field.
    3. readonly struct- a readonly indicates that the structure type is immutable.
    4. ref readonly method return, the readonly modifier indicates that method return a reference and writes are not allowed to that reference.
1. *Const*- Keyword: const. - declared a constant field or constant local. Constant fields and locals aren't variables and may not be modified. Constants can be numbers, Boolean values, strings, or a null reference. Don’t create a constant to represent information that you expect to change at any time. For example, don’t use a constant field to store the price of a service, a product version number, or the brand name of a company. These values can change over time, and because compilers propagate constants, other code compiled with your libraries will have to be recompiled to see the changes. Constants can be numbers, Boolean variables, strings, or null references.

1. *new*
2. *Sealed*
3. *Static*
4. *Override*
5. *Abstract*
6. *Async*
7. *Event*
8. *Extern*
9. *In (generic modifier)*
10. *New (generic modifier)*
11. *Out(generic modifier)*

12. *Unsafe*
13. *Virtual -* virtual keyword is used to specify a method that can be overridden in a derived class.
14. *Volatile*

- *Access Modifiers*
    1. *Public*
    2. *Protected*
    3. *Internal*
    4. *Private*

Keyword: *this*- is an access key that is used to access members of a base class from within a derived class.
Keyword: *base* - is used to call a method from the Parent class.

Managed code -anything part of our project being checked against CLR.
Unmanaged code - C/C++ are examples of unmanaged code. anything that isn't fitting the condition. When we open an external file. Memory cod enot

Field- one piece of data, like a variable, attached to the class (static) or each instance of the class (non-static)
Property - halfway between method and a field, from an external point of view, it's like a field (read and write it like a field). Internally, it's more like a pair of methods for get and set.
Auto-property - (Automatically- implemented property) There is a hidden private field behind this that actually stores the data. Auto-property syntax example = public string Number { get; set; }
Folder=Directory
Strings -> Combinations of characters.
Nano - is Bash's default text editor.
#- gives instructions to bash.
Casting-want to be able to use different implementation same right general code that can take many different implementations and use them. A way of polymorphism.
Keyword: as
Downcasting- keyword: is
ArgumentNullException
DateTime.<Now>-Built-in datatype
Constructor - responsible for initializing an object.

SQL- Structured Query Language
(commands, clauses)***
Anatomy of SQL
*try to create table in order to avoid so much code
Create - create a type of structure/resource/object
Table - references a table
Book - the name of the object to be created.
we are using SQL server
CREATE TABLE Books (verb noun <name>)

```
(
    Title VARCHAR(250) PRIMARY KEY,
    Author VARCHAR (100) NOT NULL,
    Pages INT NOT NULL,
    Thickness VARCHAR (10) NOT NULL,
    GenreID INT NOT NULL,
    PublisherID INT NOT NULL,
)
CREATE TABLE Genres
(
    ID INT PRIMARY KEY,
    Genre VARCHAR(100) NOT NULL
)

ALTER TABLE Genes ADD Genre VARCHAR (100) NOT NULL;
```

Structured Query Language (SQL) - A language that allows for the creation, modification, querying, and removal of data from a structured database.
Data Definition Language (DDL) - A set of SQL commands pertaining to the creation and structuring of a database and its schema. DDL does not modify the data only the structure, DDL commands are not normally used by the "end-user", who should be accessing the database through an application. Healp create and stucte databsr.Example anytime we made a table, moved field and nmade a new table that's somsething DDL would do.
Data Manipulation Language (DML) - A set of SQL command focused on the entry and modification of data in a database. DML does not modify a database schema, only the data contained within. When inserting order matters.
Data Query Language (DQL) - "SELECT" could be part of DML*
DCL/TCL-
Data
Database-
Table-
Relational Data Base(RDB)-
Relational Data Base Management System(RDBMS)-
Normalization-
Normal Forms-

- *1NF* - unique entry, atomic values
- *2NF* -
- *3NF* -

Multiplicity -attribute of a relationship specifies the cardinality or number of instances of an EntityType that can be associated with the instances of another EntityType. The possible types of multiplicity are as follows:

1. *One-to-many* -
2. *Zero/one-to-one* -
3. *Zero/one-to-many* -

Query - anytime we send to or request information from a database.
-- to make comments in SQL
Transaction Principles Atomicity Consistency Isolation Durable (ACID) -

1. *Atomicity* - (issues that can come up in the absence of a full isolation)
2. *Consistency* - not allowed to violate database constraints
3. *Isolation Levels (ODBC)*
    1. *Dirty read-* When a transaction sees uncommitted data of another transaction
    2. *Non-repeatable read*- when a transaction can read the same record/enrty/row twice and retrieve different data each time, because a different transaction committed and changed the dst of that record.
    3. *Phantom read* - When you run the same query twice and get different/more information the second time, because a other transaction committed date *between* queries.
1. *Durable* -transaction isn't DONE until it's written to permanent (not-volatile) storage.

SQL CONSTRAINTS -Rules enforced on a data columns of a table, these are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the database. Constraints can be at the column which only applies to one column and on the table level which applies to the whole table.

1. *SQL NOT NULL* -The column value cannot be empty.
2. *UNIQUIE* - The column cannot duplicate values. All values in the column must be different.
3. *PRIMARY KEY* - Uniquely identifies each row/record in a database table.
4. *FORGIEN KEY* -Uniquely identifies each row/record in any of the given database table.
5. *CHECK* - ensures that all the values in a column satisfies certain conditions.
6. *DEFAULT* -
7. *CREATE INDEX* -
8. *IDENTITY* -

Composite Key- is when we have more than one field more than one column that is going to be required to reference a single entry. Like for example format price table.
JOIN- A way to reach from one table into another and collect information across tables. Return more data from other tables with JOINs

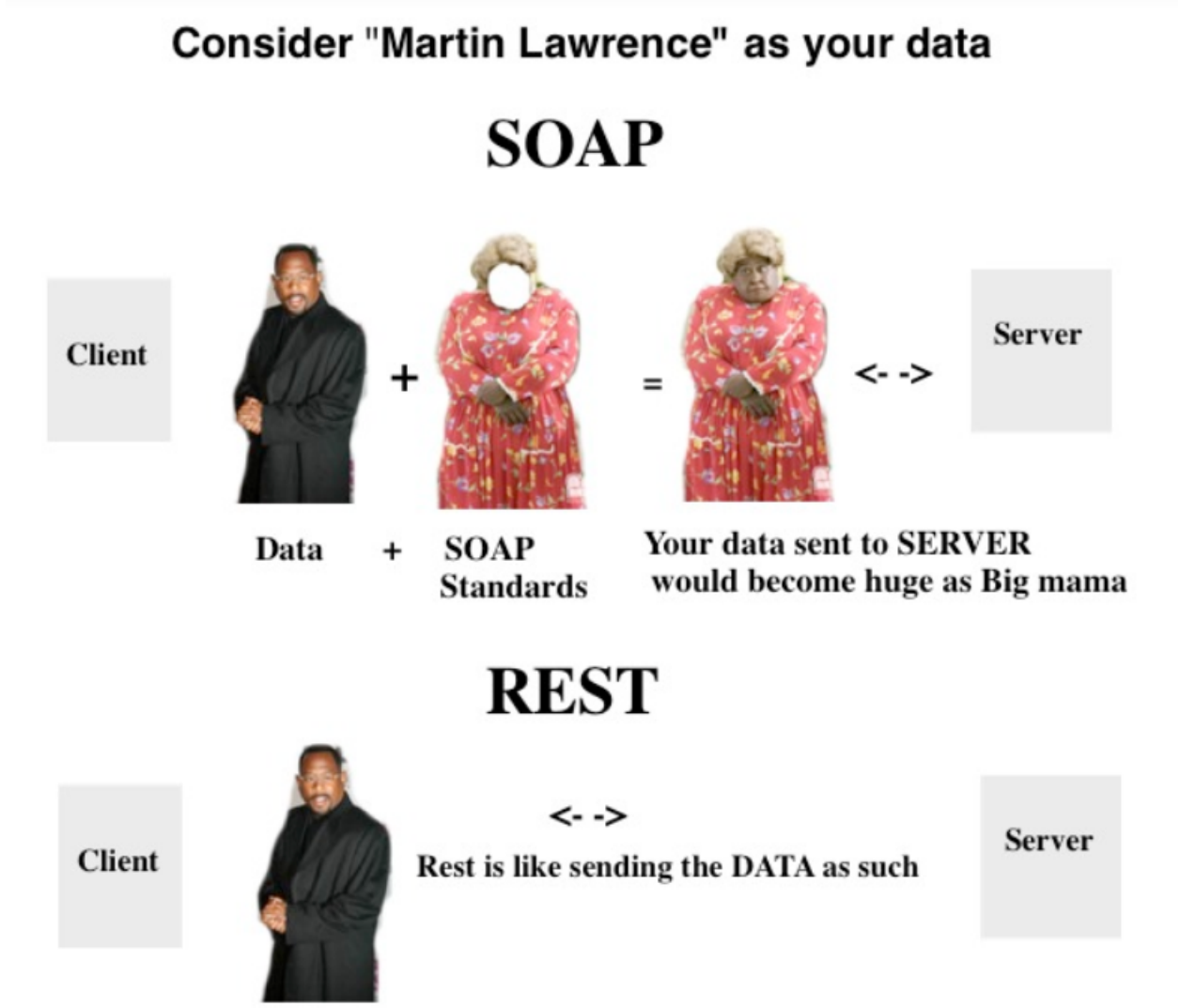
1. *INNER JOIN* - returns records that have matching values in both tables.
2. *LEFT JOIN* -returns all records from the left table, and the matched records from the right table.
3. *RIGHT JOIN* - Returns all records from the right table, and the matched records from the left table.
4. *FULL JOIN* -Returns all records when there is a match in either left or right table.
5. *SELF JOIN* - Table joins with itself

schema - like pseudocode, an object in the database that acts like a namespace. The default schema is "dbo".
Procedures
Functions

TRIGGERS

CREATE TRIGGER - Inside a trigger you have access to two table-valued variables: Insert & Deleted. These show the inserted row or the deleted row, or if its an update, Delete will have the old version of the rows, and the Inserted will have the new versions.
THROW 5000, 'message', int stage number 1
SQL Injection -
Transaction-
Guid
Hi-lo sequence
Service Oriented Architecture (SOA) -

1. *SOAP* -
2. *REST* -
    a. *Uniform Interface* - Defines the interface between **client** and **server.** It simplifies and decomposes the architecture which enables every part to be developed. Uniform interface has four guiding principles.
        i. *Resource-based:* Individual resources are identified using the URI as a resource identifier. The resources themselves are different from the representations returned to the customer. For example, the server cannot send the database but represents some database records expressed to **HTML, XML** or **JSON** depending on the server request and the implementation details.
        ii. *Manipulation of resources by representation:* When a client represents a resource associated with metadata, there is information on the server to modify or delete it.
        iii. Self-Descriptive Message: Each message contains enough information to describe how the message is processed. For example, the parser can be specified by the Internet media type (known as the MIME type).
        iv. *As the engine of Hypermedia Application State (HATEOAS):* Customers provide states by query-string parameters, body content, request headers, and requested URIs. The services provide customers with the state by response codes, response headers and body content. It is called hypermedia (hyperlink within hypertext).
    b. *Client-Server* - Servers and clients may also be replaced and developed independently, as long as the interface between them is not altered.
    c. *Statelessness* - Each request must be atomic.
    d. *Layered System* - A client cannot ordinarily tell if it is connected to the end server, or an intermediary along the way.
    e. *Cacheable* - Well-managed caching partially or completely eliminates
    f. *Code on Demand* - We are allowed extension of functionality by downloading scripts.



P1 - Establish a Connection
1 - Client establishing a connection
Usually by default on port :80
P2 - Clients sends its request and waits for the answer
-URLs contain both domain name and port
P3 - The server processes the request and replies
-sending back a body of information (data) and a status code.
-5 types/families of status code
-100s informational response
-200s -Successful responses
-300s - redirects
-400s - client errors. Like page not found
-500s - server errors. Like its getting overloaded.

HTTP request methods (case sensitive, they must be uppercase)

(Safe methods have no actions on server)

1. *GET:*- Used when the client is requesting a resource on the Web server. Its simple request for information. This method should only retrieve data and should not result in modification of data.
2. *HEAD:*- Used when the client is requesting some information about a resource but not requesting the resource itself.
3. *POST:*- Used when the client is sending information or data to the server—for example, filling out an online form (i.e. Sends a large amount of complex data to the Web Server). Modify the underlying data, create new resources
4. *PUT:*- Used when the client is sending a replacement document or uploading a new document to the Web server under the request URL. Are also used to modify data except its updating existing resources.
5. *DELETE*:- Used when the client is trying to delete a document from the Web server, identified by the request URL.
6. *TRACE:*- Used when the client is asking the available proxies or intermediate servers changing the request to announce themselves.
7. *OPTIONS:-* Used when the client wants to determine other available methods to retrieve or process a document on the Web server.
8. *CONNECT:*- Used when the client wants to establish a transparent connection to a remote host, usually to facilitate SSL-encrypted communication (HTTPS) through an HTTP proxy.
9. *PATCH* - Applies partial modification to a resource.

HTTP Method	Safe	Idempotent
GET	Yes	Yes
HEAD	Yes	Yes
OPTIONS	Yes	Yes
TRACE	Yes	Yes
PUT	No	Yes
DELETE	No	Yes
POST	No	No
PATCH	No	No

Async -

await

Task

User Interface (UI) - how the user interacts with the code

Application Programming Interface (API) - how code interacts with code

---

QC & Quizess

QC

1. What is root directory?
    1. Root directory which is referred to as / (a slash) is the topmost level of the system drive while Home directory which is /Users/<short username> (also referred to as ~) comes under the root directory. In root directory, the admin has the access for any changes in the configuration settings whereas if any user has only access to home directory, then he wont be able to change configurations of the entire system.
2. What is bash shell?
3. How would you create a bash script?
4. Difference between relative and absolute file path?
5. Where are root and home directories located?
6. What is shebang syntax !#??
7. What command is used to search for text in a file?
8. Difference between terminal and shell?
    1. terminal = text input/output environment
    2. console = physical terminal
    3. shell = command line interpreter
9. Define chmod, how would you change a file to read only?
10. What is the purpose of scripting?
11. How are cd and chmod different?
12. How can you edit a file in Bash?
13. What is nano and vim?
14. What is ssh?
15. Difference between ls and mkdir?
16. Difference between touch and cat command?
17. What is C#?
    1. An Object-Oriented programming language. It's a general-purpose meaning there are many applications you can build using C# for example console applications Desktop applications, web services and applications and much more. It is a multi-paradigm programming language[still trying to figure that part out]. The created of C# took the best parts of java, c, c++ and innovated by introducing new concepts like types, properties and events. Another benefit of C# is garbage collector, So for example users don't need to worry about memory management.
18. What is .NET?
    1. The .NET framework applications are multi-platform applications. The framework has been designed in such a way that it can be used from any of the following languages: C#, C++, Visual Basic, Jscript, COBOL, etc. All these languages can

    access the framework as well as they communicate with each other. It can help users write Windows applications web applications and web services.
19. Difference between explicit and implicit types in C#
20. What is whoami
21. What is a way to share code with multiple people.
    1. GITHUB
22. What's the difference between git and GitHub.
23. Why even use shell script.
24. Give examples of C# values.
25. Difference between explicit and implicit type conversion.
26. What is Abstraction?
    1. a way to cut out all the data and simplify code. Like Facebook or Instagram. You take a picture or write out some text and press post. Abstract is the behind the scenes of what it takes to post the user's content.
27. What is Polymorphism in C#?
    1. Polymorphism, in C#, is the ability of objects of different types to provide a unique interface for different implementations of methods. ... Polymorphism forms one of the fundamental concepts of object-oriented programming, along with encapsulation and inheritance. Polymorphism provides the ability to a class to have multiple implementations with the same name. has one name with multiple functionalities.
28. What is an Inheritance?
    1. In C#, inheritance is a process in which one object acquires all the properties and behaviors of its parent object automatically. In such way, you can reuse, extend or modify the attributes and behaviors which is defined in other class. Include I in the inheritance class to know its Inherit. In C#, the class which inherits the members of another class is called derived class and the class whose members are inherited is called base class. The derived class is the specialized class for the base class. The code can be reusable, less code will be required since it can reuse members of the parent class.
29. What is Encapsulation?
30. A way to protect data.
31. What is SHH?
    1. SSH or Secure Shell is a network communication protocol that enables two computers to communicate (c.f http or hypertext transfer protocol, which is the protocol used to transfer hypertext such as web pages) and share data.
32. Difference between Primitive Types and Reference Types.
    1. Primitive hold the value of the primitive type directly.
    2. Reference hold reference pointer to objects which reside in the garbage collection heap.
33. What's a list?
    1. Like an array but not fixed. keywork is List <T>
34. What is an array?

    1. In C#, array is an object of base type System.Array. In C#, array index starts from 0. We can store only fixed set of elements in C# array. Array can be string or integer but not mixed. To create a single dimensional array, you need to use [] after the type. Example int[] arr = new int[5]; To declare and initialize do int[] arr = new int [5]{10,20,40} or int[] arr = new int []{10,20,40} or int[] arr = {10,20,40}
35. What's your favorite OOP Pillar.
36. Name access modifiers and explain.
37. How to make read only and write only modifiers.
38. What is the difference between list and array?
39. What is polymorphism in C#?
40. Method overloading.
41. what the difference between exceptions and errors.
42. what a method, class, and constructor.
43. What is CTS?
44. What is unit testing?
45. What is CRL?
46. What is the difference between generic and non-generic collections?
47. What is boxing and unboxing?
    1. Boxing upcasting, casting to a supertype, implicit type casting
    2. Unboxing downcasting casting to a subtype, explicit type casting.
48. What is Serialization
49. What is CTS (Common Type System)?
50. What is the I in SOLID?
51. What is unit testing? What is the D in SOLID?
52. What does CLR do for us?
53. What's the difference between Generic and non-Generic collections?
54. What is boxing/unboxing?
55. What is serialization and how do we use it?
56. What is the S in SOLID?
57. How does garbage collection work?
58. Where do reference types and value types get stored?
59. How do you deal with unmanaged code and what is managed code?
60. What is a nullable data type?
61. What is the syntax to make something nullable?
62. What's the difference between fact and theory in unit testing?
63. What are the three A's to follow in unit testing?
64. What is the O in SOLID?
65. What is the CIL?
66. What is a namespace?
67. What is SOA and why use it?
68. Elaborate on loose coupling
69. What does it mean to be a service?
70. What is one of the implementation of SOA is REST? What can you tell me about REST.
71. What is HTTP?

72. What is the life cycle of HTTP?
73. What are some SOA principles
74. What are the uniform constraints?
75. What is the difference between authentication and authorization.
76. What is idempotent?
77. What can you tell me about controllers?
78. What is the ASP.NET web lifecycle
79. What are the three main parts that make a HTTP request.
80.

Quizzes

W2 Quiz Week A Assessment

1. Which command can be used to rename a file?
    1. mv
2. Which command will run the output of one command as input for a second command?
    1. |
3. Which command redirects the output of a command from stdout to a file, overwriting it if the file already exists?
    1. >
4. Bash is a shell program that emulates a command-line terminal interface and interprets user commands.
    1. True
5. What command could I execute to go up one directory lever and then search the contents of that directory for all .txt files?
    1. cd ../ && ls | grep ".txt"
6. How to create a directory in Unix System?
    1. mkdir directoryname
7. What does this command do ? Echo "hello world" >file1.txt
    1. Writes the words "hello world" into file1, overriding the existing content of the file.
8. How would I search a file name "myfile.txt" for the word "Unix"
    1. grep myfile.txt "Unix"
9. What is the command for recursively deleting files in a directory?
    1. rm -r
10. What does the "rm" command do?
    1. Deletes a file.
11. What does the "mv" command do?
    1. Moves a file from one location to another.
12. What is git?
    1. A version control tool

13. Running this Git command can fail if there are conflicting local changes?
    1. git pull
14. These Git commands are still useful even if you have no changes of your own to make to the repo.
    1. git clone
    2. git pull
15. Oder the operations for the sensible workflow in Git.
    1. <code>git clone </code>
    2. Make changes to file
    3. <code>git add</code>
    4. <code> git commit</code>
    5. <code>git push</code>
16. In bash, **~** represents what?
    1. The current user's home directory.
17. In bash, **./** represents the current directory and **../** represents the parent directory (opne level up)
    1. True
18. Which Unix commands would be useful to investigate the contents of a file?
    1. cat
    2. grep
19. Which kind of path has the same meaning regardless of the current directory?
    1. Absolute path
20. The **touch** command in Unix can be used to create new files and directories.
    1. False
21. Which Unix command will show the contents of the parent directory?
    1. ls ..
22. Which of the following is not a value type?
    1. Array
23. Which of the following keyword will terminate a while loop?
    1. break.
24. Which of the following control statements is incorrect?
    1. If (conditon1) else {//some code} else if (condition2) {//some code}
25. Which of the following are the correct ways to increment the values of variable by 1?
    1. 2 - a+=1;
    2. 4 - a=a+1;
26. Which of the following statements correctly define .NET Framework?
    1. It is an environment for developing, building, deploying and executing Desktop Applications, Web Applications, and Web Services.
27. Which operator is used in order to increment a number by 1?
    1. ++
28. The modulus "%" return the remainder of two int values when divided?
    1. True
29. Which operator returns a bool value?
    1. ||

30. You can create an if statement without an else clause?
    1. True
31. Which data types deal with numerical values?
    1. Double
32. In an if-else statement, if the conditional value in the if statement is true, then the code in the else statement will run.
    1. False
33. The decimal data type has a greater memory footprint than a double.
    1. True
34. If you pass in a value that will always be true in a while loop you will experience an infinite loop.
    1. True
35. Which of the following are not considered operators in C#?
    1. ""
36. The int keyword is used for both whole numbers and decimal values?
    1. False
37. Which keyword is used in order to represent True or False values?
    1. Bool
38. What will be the output of the following C# code?

```
static void Main(string[] args)
{
    int i;
    int b = 8, a = 32;
    for (i = 0; i <= 10; i++)
    {
        if ((a / b * 2) == 2)
            Console.Write(i + " ");
        else if (i != 4)
            Console.Write(i + " ");
        else
            break;
    }
}
```

    1. 0 1 2 3
1. What will be the output of the following C#

```
static void Main(string[] args)
{
    int i;
    for (i = 0; i < 5; i++)
    {
    }
    Console.WriteLine(i);
}
```

1. 5
1. In C# which is the normal way to check if two strings **s1** and **s2** are equal?
    1. if (s1 == s2)
2. Which correctly declares variables in C#?
    1. Int a = 32; int b = 40;
3. .NET Framework is cross-platform, but .NET core is only for windows.
    1. False
4. What is the command **dotnet new** used for?
    1. Create a new .NET Core program

W3 Quiz OOP and Class

1. In C#, which project templates involve a **Main** method?
    1. console app
2. In C#, constructors have to be initialize every field.
    1. FALSE
3. In C# what's the meaning of **void.**
    1. The return type of a method that returns nothing.
4. Which are NOT true in C# about arrays once they are created?
    1. Their contents cannot be change.
    2. They can only contain numbers or strings.
5. Which is a good way to send data between different parts of a C# program?
    1. Method parameters and returns.
6. Given a string[] variable in C# named words, what expression will give the first character of the second string in the array?
7. In C#, variables are not accessible outside the code block in which they are declared.
    1. TRUE
8. Which correctly declares variables in C#?
    1. Int a = 32; int b = 40;
9. What is false about Interfaces in C#?
    1. They can be used to allow for multiple inheritance.
    2. All methods in an interface must be implemented.
    3. Both a class and a struct can implement them.
    4. You define an interface with the interface keyword.
    5. **All of these are true about interfaces.**
10. In C#, one class can inherit from multiple base classes.

    1. False
11. Which character is used to start and end code blocks in C#?
    1. {}
12. An If-Else statement is used for handling thrown Exceptions
    1. False , Try catch block are is used.
13. Which keyword is used for raising an exception?
    1. Throw
14. What is an Exception?
    1. Error that occurs at run-time during program execution.
15. In OOP, computer programs are designed by making them out of ___ that interact with one another.
    1. objects
16. The four pillars are__
    1. Abstraction
    2. Polymorphism
    3. Inheritance
    4. Encapsulation
17. ___ is the process by which a developer separates the relevant data from the irrelevant details in order to simplify use.
    1. Abstraction
18. __ is the restricting of direct access to abstracted data.
    1. Encapsulation
19. __ allows you to define a child class that reuses (inherits) the characteristics of a parent class.
    1. Inheritance
20. Modifiers are C# keywords used to modify declarations of ___.
    1. Types and type members
21. 'abstract' means___
    1. The thing being modified has a missing or incomplete implementation.
22. 'virtual' means__
    1. A member can be overridden in derived class.
23. When overriding a member, ___
    1. You must provide a new implementation of an inherited method.
24. Which access modifiers would be used if you want accessibility within the same class only.
    1. Private
25. What is the default access level of class members?
    1. Private
26. Which core Object Oriented Programming principle allows you to generalize and specialize a code in combination with inheritance?
    1. Polymorphism
27. What will you use in order to mimic multiple inheritance?
    1. Interfaces

28. Given the following class and interface signatures, which is the best way to apply multiple inheritance to DerivedRobot class?

```
interface Irobot2 { //code }
interface IrobotB { //code }
class Robot1 { //code }
class RobotA { //code }
class DerivedRobot { //code }
```

1. Class Robot1 : RobotA {//code} class DerivedRobot : RobotA, Irobot2, IrobotB{//code}
1. Which of the following can be facilitated by the Inheritance mechanism?

1. Use the existing functionality of base class.
2. Overrride the existing functionality of base class.
3. Implement new functionality in the derived class.
4. Implement polymorphic behaviour.
5. Implement containership.

1. 1,2,3
1. Object and reference types are stored on the Stack
    1. False
2. Which namespace or base library are all value types derived from?
    1. System.Object
3. The default constructor can be overridden
    1. True
4. What keyword is used in order to create an instance of a class?
    1. new
5. You can only have one constructor.
    1. False
6. A class can have no constructors.
    1. False
7. All values types are stored on the stack.
    1. False
8. Dictionary is a generic version of __>
    1. Hashtable
9. When you create an instance of a generic class, you specify the__ to substitute for the type parameters.
    1. types
10. What property is used to get the number of items in an array?
    1. Length
11. What collection type uses keys, value pairs?
    1. Dictionary

W4 Quiz C#, .NET, Intermediate Git

1. What is false about Interfaces in C#? (Doubled)
    1. They can be used to allow for multiple inheritance.
    2. All methods in an interface must be implemented.

3. Both a class and a struct can implement them.
4. You define an interface with the interface keyword.
5. **All of these are true about interfaces.**

2. Is C# a strongly typed language, what is a strongly typed language?
    1. Yes, each type of data is predefined as part of the programming language.
3. What is garbage collection?
    1. The process of de-allocating memory automatically.
4. An If-Else statement is used for handling thrown Exceptions. (Doubled)
    1. False
5. Which of the following statements are correct about the C# .NET code snippet given below?

```
namespace IndiabixConsoleApplication
{
    class index
    {
        protected int count;
        public index()
        {
            count = 0;
        }
    }
    class index1: index
    {
        public void increment()
        {
            count = count +1;
        }
    }
    class MyProgram
    {
        static void Main(string[] args)
        {
            index1 i = new index1();
            i.increment();
        }
    }
}
```

1. *count* should be declared as *public* if it is to become available in the inheritance chain.
2. *count* should be declared as *protected* if it is to become available in the inheritance chain.
3. While constructing an object referred to by *i* firstly constructor of index class will be called followed by constructor of *index1* class.
4. Constructor of index class does not get inherited in *index1* class.
5. *count* should be declared as Friend if it is to become available in the inheritance chain.

1. 2, 3, 4

1. How do you access a private field that is a member of a class?
    1. With a public property to get and set the private field.

2. What is the correct output of the following code sample?

```
    namespace eIntern {
    class Robot {
        string type;
        double speed;
        public void SetRobot(double speed) {
            type = "hover";
            speed = speed;
        }
        public void Output() {
            Console.WriteLine(type +"bot has speed " +
speed);
        }
    }
    class Program {
        static void Main(string[] args) {
            Robot droid = new Robot();
            droid.SetRobot(32);
            droid.Output();
        }
    }
}
```

1. Error will prevent code from executing.

1. Which of the following is an incorrect signature for a constructor for a Robot class?
    1. public void Robot(int speed)
2. What is the correct way to declare and instantiate a List collection?
    1. List<int> numbers = new List<int>();
3. Provide a List with 10 names, which is the correct way to get the name at 5th position in the List?
    1. names[4];
4. Given the following class and interface signatures, which is the best way to apply multiple inheritance to DerivedRobot class?

```
interface Irobot2 { //code }
interface IrobotB { //code }
class Robot1 { //code }
class RobotA { //code }
class DerivedRobot { //code }
```

1. Class Robot1 : RobotA {//code} class DerivedRobot : RobotA, Irobot2, IrobotB{//code}
1. Which keyword is used to add a base class library or namespace to a project to become accessible?
    1. using
2. Of the following components, which cannot be inside a class definition?
    1. namespace
3. What do yo use as a flag or marker indicating a place to pause current execution of a program?
    1. Breakpoint
4. Which keyword is used to call a method from the Parent class?

    1. Base
5. What will you use in order to mimic multiple inheritance?
    1. Interfaces
6. Objects and Reference types are stored on the Stack.
    1. False, Objects and Reference types are stored on the Heap.
7. All value types are stored on the stack.
    1. False, value types that are stored in an object as a field or property are stored on the Heap with the object.
8. Which of the following .NET components can be used to remove unused refences from the managed heap?
    1. Garbage Collector
9. Which of the following statements is correct about Managed code?
    1. Managed code is the code that is written to target the services of the CLR.
10. What command changes your current branch?
    1. git checkout
11. What is the name of the branch that is created by default when a new repository is made?
    1. master or main
12. In Git, after making changes in a branch, you can push without committing.
    1. false
13. Which of the below git command can create a new branch in Git?
    1. git checkout -b <newBranch>
14. What is a protected branch in Git?
    1. A branch that cannot be accidentally deleted
    2. A branch that you cannot push into unless you are the admin or owner
    3. A branch in which changes can only go in with a pull request approved by admin
    4. All of the options

W5 Quiz SQL Weekly Assessment

1. What do we call a primary key that consists of multiple columns in a table?
    1. Composite key
2. The UNIQUE constraint requires that values must be
    1. Unique in the column
3. Referential integrity stipulates
    1. Foreign keys must always reference a valid unique key
4. The CHECK constraint is used to
    1. Require any values placed in a column to satisfy a logical expression
5. The ON DELETE clause is used to
    1. Specify behavior to take when a referenced row is deleted
6. A foreign key is
    1. A constraint which defines that a column reference a primary key in another table or row.
7. Constraints
    1. Used to restrict values that can be placed in a column
8. A primary key is

    1. Implicitly UNIQUE
    2. Implicitly NOT NULL
    3. A value used to uniquely identify a row
9. MAX(), SUM(), and COUNT() are examples of
    1. Aggregate functions
10. UPPER(), ABS(), CONCAT() are examples of
    1. Scalar functions
11. What is required of primary key?(Select all that apply)
    1. Unique Value
    2. Not null
12. What do we call a field that references a field in another table?
    1. Foreign Key
13. Which of the following joints is best used to return only records that meet a condition?
    1. Inner join
14. Which of the following joins will return all records from the first table despite any conditions specified?
    1. Left join
15. Which of the following joins will return all records from both tables despite any conditions specified?
    1. Full join
16. Use the AS keywords to specify an alias
    1. True
17. CRUD stands for:
    1. Create , Read, Update, Delete
18. Which of the following clauses are added to sort the results of a SELECT statements?
    1. ORDER BY
19. A subquery is a SQL query that inserts a record into a table.
    1. False
20. DML statements include commands like CREATE, DROP, and ALTER
    1. False
21. DDL is a subset of SQL that deals with data creation and includes commands like CREATE
    1. True
22. In a many-to-many relationship, you typically use a third table to manage the actual associations between rows in one table and another.
    1. True
23. In a one-to-many relationship, rows in one table can refer to multiple rows in another, but that other table can only refer to at most one row in the former table
    1. True
24. In a one-to-one relationship, a record in one table can at most refer to one record in another table.
    1. True
25. A constraint is applied to a row and not a column
    1. False

26. SQL stands for:
    1. Structured Query Language
27. What is not a use of the WHERE clause?
    1. WHERE locates which database/schema/table where a row is located
28. What is the difference between UNION and UNION ALL?
    1. UNION returns only distinct rows, while UNION ALL returns all rows
29. The SQL UPDATE statement can..
    1. Update multiple rows at a time
30. What is SQL Injection?
    1. Injecting a SQL Statement as user input
31. What does the CREATE TABLE statement do?
    1. Creates a new database table
32. Constraints are used to specify rules for the data in a table.
    1. True
33. The Primary Key and Foreign Key constraints allow for the relationships to be created between tables.
    1. True
34. What symbol is use to represent everything when selected from a table.
    1. *
35. What SQL statement is used to update data in a database?
    1. update
36. Which SQL statement is used to delete data from a database?
    1. delete
37. Which SQL statement is used to insert new data in a database?
    1. insert into
38. Which of the following SQL statements will select all records with all their columns from a table called Sales.
    1. SELECT*FROM Sales
39. How do you drop a table?
    1. DROP TABLE table_name
40. What is the difference between Order By and Group By?
    1. ORDER BY is used for sorting results, whereas GROUP BY is used with aggregate functions to group results

W6 Quiz Weekly Assessment

1. What are benefits of SOAP over REST?
    1. SOAP services are accompanied by a clearly outlines the service
    2. SOAP services are not bound to a specific protocol
    3. SOPA services can use a variety of data formats
    4. All of the above
2. What not true about service-oriented architecture (SOA)?
    1. Black box services that orchestrate into an entire ecosystem
    2. Web services are the preferred technical approach to achieve SOA
    3. Services should be loosely coupled by hiding the service implementation
    4. None of the above

3. Why are Web Services needed?
    1. Provides interoperability between platforms
    2. Promotes code reusability
    3. Allows for distributed business applications
    4. All of the above
4. What are Web Services?
    1. Software that makes itself available over the Internet and uses a standardized messaging system
    2. Hardware that can be made accessible over the Cloud
    3. Software that runs on a local platform but mimics a Web browser interface
    4. All of the above
5. SOAP messages contain _____; and REST messages contain _____.
    1. XML, XML
    2. XML or JSON, XML or JSON
    3. XML, JSON
    4. XML, XML or JSON
6. What transport protocol is used in REST?
    1. SMTP
    2. HTTP
    3. FTP
    4. All of the above
7. What language is used to represent a resource?
    1. Plain Text
    2. XML
    3. JSON
    4. All of the above
8. What is a resource in REST?
    1. Any content accessible through the Web Service
    2. A socket connection to access the Web Service
    3. A database connection object for the service to access its data
    4. All of the above
9. Which is not a commonly used HTTP method in REST?
    1. GET
    2. PUT
    3. POST
    4. DELETE
    5. TRACE
    6. OPTIONS
10. Which is not one of the core components of an HTTP request?
    1. HTTP method
    2. Uniform resource identifier (URI)
    3. Status code
    4. Request Body
    5. Request Header

11. Which is not one of the core components of an HTTP response?
    1. Response Header
    2. Response Body
    3. HTTP Version
    4. Status Code
    5. URI
12. The _____ HTTP status code shows success in creating something.
    1. 500
    2. 403
    3. 201
    4. 200
13. How are exceptions handled in REST?
    1. A web service should pass back an Exception object to the client
    2. A web service should use HTTP error codes like 403 to show access forbidden using the Response object
    3. Using faults like the way SOAP does
    4. Exceptions should not be sent back to the client
14. The purpose of a _____ is to locate a resource on the server hosting the REST web service.
    1. Service endpoint
    2. Uniform resource identifier (URI)
    3. Request parameter
    4. SOAP message
15. The _____ HTTP status code shows not found when a resource is unavailable.
    1. 500
    2. 403
    3. 404
    4. 200
16. The _____ HTTP status code shows internal server error.
    1. 500
    2. 403
    3. 404
    4. 200
17. The _____ HTTP status code shows forbidden access to a resource.
    1. 500
    2. 403
    3. 404
    4. 200
18. The _____ HTTP status code shows a bad request.
    1. 400
    2. 403
    3. 500
    4. 200
19. Which HTTP method can be used for creating or inserting new data onto a server?

    1. TRACE
    2. PUT
    3. POST
    4. CREATE
20. Which character indicates the beginning of a query param in a URL?
    1. ?
    2. /
    3. ;
    4. :
21. What are the different types of web services?
    1. Soap
    2. Rest
    3. Soa
    4. Remote
22. Which of the following is correct about URI in RESTful web services?
    1. Each resource in REST architecture is identified by its URI.
    2. Purpose of an URI is to locate a resource(s) on the server hosting the web service.
    3. Both of the above.
        1. Explanation : Each resource in REST architecture is identified by its URI. Purpose of an URI is to locate a resource(s) on the server hosting the web service.
    4. None of the above.
23. Which of the following is correct about statelessness in context of RESTful web service?
    1. As per REST architecture, a RESTful web service should not keep a client state on server.
    2. It is responsibility of the client to pass its context to server and then server can store this context to process client's further request.
    3. Both of the above.
        1. Explanation : Both of the above options are correct.
    4. None of the above.
24. Applications can communicate using XML and still be considered RESTful:
    1. True
    2. False
25. What is true about the Layered System principle of REST? Select one of the following:
    1. According to this principle, any one service cannot see beyond its own functionality.
    2. According to this principle, there is always a service above and below the current service.
    3. There is no Layered System principle of REST.
    4. According to this principle, there should only by one service.
26. What is contained in an HTTP Request?
    1. Select all that apply from the following:
    2. Headers containing metadata

    3. A verb
    4. A body, potentially containing data
    5. A destination URL
    6. A response
    7. An HTML form
27. HTTP is stateless
    1. TRUE
    2. FALSE
28. Which HTTP Verb is used in order to send data to the server?
    1. GET
    2. POST
    3. PUT
    4. DELETE
29. Which HTTP Verb is used in order to edit data that is currently on the server?
    1. GET
    2. POST
    3. PUT
    4. DELETE
30. Which HTTP Verb is used in order to remove data that is currently on the server?
    1. GET
    2. POST
    3. PUT
    4. DELETE
31. What do you use in order to send information or other data along with the request that is needed by the server?
    1. Request Headers
    2. URI
    3. HTTP Services
32. Which Request Header is used in order to specify what type of data is expected with the response?
    1. Accept Header
    2. Content-type Header
    3. Allow Header
33. Which request header is used when you send values from a form so that the server knows what type of data to expect with the request?
    1. Accept Header
    2. Content-type Header
    3. Send Header
34. In a HTTP Service, where is the data such as JSON or XML stored in a HTTP Response?
    1. Body
    2. Header
    3. Status Code
    4. Content-type header

35. Serializing is not included with ASP.NET Web API.
    1. TRUE
    2. FALSE
36. ASP.NET Web API will automatically bind your JSON or XML files to regular C# classes.
    1. TRUE
    2. FALSE
37. You can use Scaffolded Controllers which will create method signatures for the 4 verbs available for Web API
    1. TRUE
    2. FALSE
38. HttpClient is part of what namespace?
    1. System.Net.Http
    2. Newtonsoft.Json
    3. System.Runtime
    4. System.Http
39. Which of the following statements are true about REST?
    1. REST is web standards based architecture and uses HTTP Protocol for data communication.
    2. It revolves around resource where every component is a resource and a resource is accessed by a common interface using HTTP standard methods
    3. REST was first introduced by Roy Fielding in 2000.
    4. All of the above.
40. Which of the following is correct about addressing in RESTful web services?
    1. Addressing refers to locating a resource or multiple resources lying on the server.
    2. It is analogous to locate a postal address of a person.
    3. Both of the above.
    4. Explanation : Addressing refers to locating a resource or multiple resources lying on the server. It is analogous to locate a postal address of a person.
    5. None of the above.
41. Which REST annotation specifies the Media Type that the resource returns to the client?
    1. Path
    2. Produces
    3. Consumes
    4. Request
42. Which REST annotation specifies the Media Type that the client can send to the resource?
    1. Path
    2. Produces
    3. Consumes
    4. Request
43. True or False: REST is used to implement a client-service architecture
    1. TRUE
    2. FALSE

44. Which of the following HTTP method should be used to fetch resource using RESTful web service?
    1. GET
        1. Explanation : GET operations should be used to fetch resource using RESTful web service.
    2. DELETE
    3. PUT
    4. OPTIONS
45. Which of the following HTTP method should be used to get status of method availability in RESTful web service?
    1. GET
    2. HEAD
        1. Explanation : HEAD operations should be used to get status of method availability in RESTful web service.
    3. PUT
    4. OPTIONS
46. Which of the following HTTP method should be used to get list the supported operations in RESTful web service?
    1. GET
    2. HEAD
    3. PUT
    4. OPTIONS
47. Which of the following is true about caching in RESTful web service?
    1. Caching refers to storing server response in client itself so that a client needs not to make server request for same resource again and again.
    2. A server response should have information about how a caching is to be done so that a client caches response for a period of time or never caches the server response.
    3. Both of the above.
        1. Explanation : Both of the above options are correct.
    4. None of the above.
48. Which of the following approaches is considered best practice for designing a secure RESTful web service?
    1. No sensitive data in URL - Never use username, password or session token in URL , these values should be passed to Web Service via POST method.
    2. Restriction on Method execution - Allow restricted use of methods like GET, POST, DELETE. GET method should not be able to delete data.
    3. Both of the above.
        1. Explanation : Both of the above options are correct.
    4. None of the above.
49. Which of the following is a best practice for designing a secure RESTful web service?
    1. Validate Malformed XML/JSON - Check for well formed input passed to a web service method.

2. Throw generic Error Messages - A web service method should use HTTP error messages like 403 to show access forbidden etc.
3. Both of the above.
    1. Explanation : Both of the above options are correct.
4. None of the above.

50. REST is web standards based architecture and uses HTTP Protocol for data communication.
    1. false
    2. true
51. Which of the following is correct about resource representation in REST?
    1. REST uses various representations to represent a resource where text, JSON, XML.
    2. XML and JSON are the most popular representations of resources.
    3. Both of the above.
        1. Explanation : REST uses various representations to represent a resource where text, JSON, XML. XML and JSON are the most popular representations of resources.
    4. None of the above.
52. Which of the following is correct about RESTful web services?
    1. Web services based on REST Architecture are known as RESTful web services.
    2. These web services use HTTP methods to implement the concept of REST architecture.
    3. A RESTful web service usually defines a URI, Uniform Resource Identifier a service, provides resource representation such as JSON and set of HTTP Methods.
    4. All of the above.
53. Define some REST characteristics that SOAP doesn't possess
    1. Utilizes HTTP only
    2. It supports multiple mediatypes
    3. It is not contract based
    4. It is contract based
    5. It supports multiple protocols
    6. Utilizes XML only
54. What does REST stand for?
    1. Representational State Transfer
    2. Representational Structured Transfer
    3. Representational System Transfer
    4. Representational State Translation
55. Which of the following is correct about messaging in RESTful web services?
    1. A client sends a message in form of a HTTP Request and server responds in form of a HTTP Response.
    2. These messages contain message data and metadata i.e. information about message itself.
    3. Both of the above.

        1. Explanation : A client sends a message in form of a HTTP Request and server responds in form of a HTTP Response. This technique is termed as Messaging. These messages contain message data and metadata i.e. information about message itself.
    4. None of the above.
56. Which of the following is advantage of RESTful web service being stateless?
    1. Web services can treat each method request independently.
    2. Web services need not to maintain client's previous interactions. It simplifies application design.
    3. As HTTP is itself a statelessness protocol, RESTful Web services work seamlessly with HTTP protocol.
    4. All of the above.
57. Which of the following is true about REST?
    1. In REST architecture, a REST Server simply provides access to resources and REST client accesses and presents the resources.
    2. Each resource is identified by URIs/ global IDs.
    3. REST uses various representations to represent a resource like text, JSON and XML.
    4. All of the above.
58. Which of the following is not a valid HTTP method used in RESTful web services?
    1. GET
    2. PUT
    3. TIME
        1. Explanation : TIME is not a HTTP Verb and is not a valid HTTP methods used in RESTful web services.
    4. POST
59. In REST architecture, a REST Server simply provides access to resources and REST client accesses and presents the resources.
    1. false
    2. true
60. RESTful web services use HTTP methods to implement the concept of REST architecture.
    1. true
        1. Explanation : RESTful web services use HTTP methods to implement the concept of REST architecture.
    2. false
61. A RESTful web service usually defines a URI, Uniform Resource Identifier a service, provides resource representation such as JSON and set of HTTP Methods.
    1. false
    2. true
62. RESTful web services make use of FTP protocol as a medium of communication between client and server.
    1. true
    2. false

63. A RESTful web service client sends a message in form of a Gopher Request and server responds in form of a HTTP Response.
    1. true
    2. false
64. URI of HTTP request indicates HTTP methods to be executed by RESTful Web services.
    1. false
        1. Explanation : HTTP Verb − Indicates HTTP methods such as GET, POST, DELETE, PUT etc.
    2. true

Commands/HowTo

- GITBASH Commands

<CurrentUser>/<Machine>MINW64~

~	(Home Directory)
Pwd	Print working (current) directory.
c ../	Up one directory
ls	List contents of current folder.
ls ./<FolderName>	List contents of specific folder.
mkdir <NameFolder>	Creates new directory.
touch <NameFile.FileExtenstion>	Creates new file, like .txt, .md, .sh, etc.
rm <Filename>	Deletes file.
rm-r <Foldername>	Deletes folder.
rm ./<Folder you need to go into>/<File in folder you want to delete. FileExtension>	Deletes specific file in a specific folder.
rm -r -i <Folder your deleting>	Asks if you are sure.
help->.	List helpful commands/flags
<Command> --help	Detailed help on command.
c ..	Parent Directory.

c ./../<Directory>/<Directory>/	Navigate through two folders in one command.
cat <Filename>	read/prints content.
clear	Clears screen.
exit	Exits.
start .	Open current directory.
start <program Ex.Notepad> <File.Extension>	Open/run a file in desired program.
explore	open a file explorer to the target location.
git	
"" clone <GitHub URL>	Creates linked repository to local.
"" add <File/Foldername>	Specifies what you want to add.
"" add .	Adds everything.
"" status	Displays the status of the repo, including files not yet added, added but not committed, and what commit the repo is on.
"" push	Sends the most recent commit to the repo.
"" pull	Pulls the most recent commit from cloud repo to the local repo.
"" restore	Restores las commit.
Echo "hello"	Prints hello
Echo "hello" > <Filename.Ext>	Prints message into file and overwrites.
echo "Hello" >> <Filename.Ext>	Prints new string, append.
which <Application>	Shows path
"" --Version	Displays version of application.
|	String together commands in the same command line.
ls -a	Shows hidden content.
ls -l	shows permissions
chmod	Allows users to change permissions -remove + add => overwrite

nano	Text editor.
grep	Search within file for words or types of files.
sed	Find word and replace (Stream Editor). Example $sed 's/<oldtext>/<newtext>g' <filename.ext>

Bash Script

#	
#!	
echo	
$	

$-is an injections. Drops values without needing to use the concatenation of a string.
Example: ${<display>}:<string> literal{<display>}

- HOW TO CLONE A REPOSITORY

Pick location/directory
mkdir <Foldername>
cd <Foldername>
git clone <Paste github URL>

- HOW TO BASH SCRIPT

Create script file
touch <Filename.sh)
open in notepad.
On top type
#!/usr/bin/bash
<now you can script>

- CREATING A VARIABLE

#!/usr/bin/bash
variable = "Hello" -> Create name "Hello"
echo $variable -> Prints variable content "Hello" $-used to call variable

- ACCEPT NEW VARIABLE

echo "Please Enter a Variable"
read variable -> Reads user input.
echo $variable -> Prints user input.
echo $variable "is what you entered" -> Prints user input and text.

- MAKE YOUR OWN REPOSITORY

!!DO NOT CLONE REPO IN ANOTHER REPO!!

-Go to organization on GitHub
-Press NEW
-Create a new repository
-Name repository
-Copy link
-In git terminal.
->create a directory
->cd to your directory
->git clone <Paste URL>
Congrats! You have created a repository!

- HOW TO UPDATE

git int
git add <.all or specific>
git commit -m "<comment>"
git push

- HOW TO PULL FROM A REPO

Pull from Trainer-code.
In git terminal make sure you are in the trainer-code directory and type git pull.

- CREATE BASH SCRIPT

touch <namefile.sh>
In shellscript (notepad)
#!/usr/bin/bash

#Create a new folder
mkdir ./testfolder

#Create a new file in that folder
touch ./testfolder/newfile.txt

#Write text to the new file
echo "this is some line of text" > ./testfolder/newfile.txt

- Create a C# .NET CONSOLE APP

In git terminal
*Choose a directory-> Create a directory
mkdir <Foldername>

cd <Foldername>
dotnet new console
code . -> Opens in VS code
dotnet build -> Displays project

Two ways of running code
dotnet ./bin/Debug/net6.0/<filename.dll>
OR
dotnet run
*And then displays code!

- HOW TO DO A BRANCH

use git
git switch - C (create) <NameOFNewBranch>
copy code to repo
git status (check if worked)
git add .
git commit - "<name>"
git push origin <name>

SQLCommands

--	Comments
SELECT	
INSERT	
CREATE	
CREATE DATABASE [NameDatabase]	creates a database
CREATE TRIGGER	
;	separates QUERIES

- HOW TO CREATE A NEW DATABASE

Target Master
CREATE DATABASE [nameDatabes]

- HOW TO DROP

DROP [namedTable/Database]

- HOW TO INTERACT WITH DATABASE

CREATE target Database

Short-cuts/Snippets

ctrl c - breaks operation
Ctrl-shift+a -In visual code to add new class do

- VISUAL STUDIO SHORTCUTS

Ctrl-Shift+a- Makes a new class
ctrl-k-k (add bookmark) ctrl-w-b -Bookmarks
Ctor (snippet) - makes a constructer
prop tab (snip) -auto implemented property -
ex public string firsnatme { get; set;}
cw tab - Console.Writeline snip
try tab - try catch snip
tryf - try finally snip
for tab-snip
ctrl+a CTLR+K CTRL+f--clean up all code
ctrl+k+/ OR /* - Add comments
ctrl+Space ->Prompts suggestions.
...-> code can be simplified
///+tab - <summatry

CODES

HACKER CHALLENGES

--LET'S ECHO--

```
echo "Hello"
```

--PERSONILIZED ECHO--

```
read name
echo "Welcome" $name
```

--LOOPING WITH NUMBERS--

```
for x in {1..50}
do
echo $x
done
```

OR

```
for ((x=1; x<=50; x+x))
do
echo $x
done
PRINTS
1
2
3
.
.
.
50
        --LOOPING AND SKIPPING--
seq 1 2 99
        OR
for i in {1..99..}
do
echo $i
done
        OR
for ((i=1; i<100; i+2))
do
echo $i
done
PRINTS
1
2
3
.
.
.
99
        --GETTING STARTED WITH CONDITIONALS--
read x
if [ "$x" == "Y" ] || [ "$x" == "y" ]; then
echo "yes"
elif ["$x" == "N" ] || [ "$x" == "n" ]; then
echo "no"
fi
        --COMPARING NUMBERS--
read x
read y
if [$x -gt $y]
then
```

```
echo "x is greater than y"
if [$x = $y]
then
echo "x is equal y"
if [$x -lt $y]
then
echo "x is less than y"
fi
        --COMPUTE THE AVERAGE--
read n
for i in $(seq 1 $n);
do
read num
sum = $((sum + num))
done
echo $sum/$n |bc - l| xargs printf "%.3f"
```

CHALLENGES IN BASHSCRIPTS

```
#Loop 1-20, append odd to file
for i in {1..20..2}; do
          echo $i >> ./testfolder/newfile.txt
done

#filepath="./testfolder/newfile.txt"

#for (( i=1; i<20; i+=2 ))
#do
#         echo $i >> $filepath
#done


#Loop 1-20, print even numbers to a single line
#for ((i=2; i<=20; i+=2));
#do
#         if [ $i -lt 20 ]; then
#         echo -n "$i, "
#         else
#         echo -n $i
#         fi
#done


for i in {2..20..2}
```

```
do

nums+=$i
if(($i<20))
then
        nums+=", "
fi
done

echo $nums >> ./testfolder/newfile.txt
```

CONDITIONAL STATEMENTS

--IF-ELSE--

```
if (CODITION)
{
do this;
}
else if (DIFFERENT CODITION)
{
do this other thing;
}
else
{
do this;
}
```

--SWITCH--

```
switch (<EXPRESSION>)
{
case 0:
statment do;
break;
case 1:
stament do;
break;
default:
when all else fails do this
break;
}
```

--LOOPS--
do
{

}
while(CODITION);

while(CONDITION)
{
do this;
}

SQL Book database Example

SQL CHINOOK Database

For each of the following exercises, provide the appropriate query.
Keep your successful queries in a chinook-queries.sql file.

1. Provide a query showing Customers (just their full names, customer ID and country) who are not in the US.
2. Provide a query only showing the Customers from Brazil.
3. Provide a query showing the Invoices of customers who are from Brazil. The resultant table should show the customer's full name, Invoice ID, Date of the invoice and billing country.
4. Provide a query showing only the Employees who are Sales Agents.

```
SELECT *
FROM Employee
WHERE Title = 'Sales Support Agent';

SELECT FirstName, LastName, Title FROM Employee WHERE Title='Sales Support Agent';

SELECT FirstName, LastName, Title
FROM Employee
WHERE Title = 'Sales Support Agent'


SELECT FirstName, LastName, Title
FROM Employee
WHERE Title='Sales Support Agent';
```

1. Provide a query showing a unique list of billing countries from the Invoice table.

```
SELECT DISTINCT BillingCountry
FROM Invoice;
```

1. Provide a query that shows the invoices associated with each sales agent. The resultant table should include the Sales Agent's full name.
2. Provide a query that shows the Invoice Total, Customer name, Country and Sale Agent name for all invoices and customers.

```
SELECT
    Total,
    Customer.FirstName,
    Customer.LastName,
    Customer.Country,
    Employee.FirstName,
    Employee.LastName
FROM Employee
INNER JOIN Customer ON Employee.EmployeeId=Customer.SupportRepId
INNER JOIN Invoice ON Customer.CustomerId=Invoice.CustomerId
WHERE Title='Sales Support Agent';
```

1. How many Invoices were there in 2009 and 2011? What are the respective total sales for each of those years?

```
SELECT COUNT(InvoiceDate) AS TotalInvoices, SUM(total) AS TotalSales
FROM Invoice
WHERE YEAR(InvoiceDate)=2009
OR YEAR(InvoiceDate)=2011;
```

1. Looking at the InvoiceLine table, provide a query that COUNTs the number of line items for Invoice ID 37.
2. Looking at the InvoiceLine table, provide a query that COUNTs the number of line items for each Invoice. HINT: GROUP BY
3. Provide a query that includes the track name with each invoice line item.
4. Provide a query that includes the purchased track name AND artist name with each invoice line item.
5. Provide a query that shows the # of invoices per country. HINT: GROUP BY
6. Provide a query that shows the total number of tracks in each playlist. The Playlist name should be included on the resultant table.
7. Provide a query that shows all the Tracks, but displays no IDs. The resultant table should include the Album name, Media type and Genre.
8. Provide a query that shows all Invoices but includes the # of invoice line items.
9. Provide a query that shows total sales made by each sales agent.
10. Which sales agent made the most in sales in 2009?
11. Which sales agent made the most in sales in 2010?
12. Which sales agent made the most in sales over all?

13. Provide a query that shows the # of customers assigned to each sales agent.
14. Provide a query that shows the total sales per country. Which country's customers spent the most?
15. Provide a query that shows the most purchased track of 2013.
16. Provide a query that shows the top 5 most purchased tracks over all.
17. Provide a query that shows the top 3 best selling artists.
18. Provide a query that shows the most purchased Media Type.
19. Provide a query that shows the number tracks purchased in all invoices that contain more than one genre.