PRACTICE SET OOPS & SE
Level 5
Session 2022-23

Question One

A) What is the meaning of inheritance in object-oriented programming, why it is useful?
B) In the following code snippet,

- x is a public member of the class “Base”
- y is a protected member of the “class “Base”
- z is a private member of the class “Base”

due to the inheritance, the x, y and z are also members of class “A”, “B”, “C” and “D”. What is their accessibility in each of the classes respectively?

Note: Possible options for accessibility are:

- *public*
- *private*
- *protected*
- *not accessible*

```
class Base
{
    public int x;
    protected int y;
    private int z;
};

class A: Base
{
};

class B: protected Base
{
};

class C : private Base
{
};
```

```
class D: public Base
{
};
```

Model Answer:

A) Inheritance in OOP means a derived class inherits all class members from its base class as the base class "gives birth" to the derived class.
It is a technique for software reuse, a derived class can reuse the code written in its base class and expand the functionality of the base class without affecting the original implementation.

B)

```
Class Base
{
   public:
      int x;
   protected:
      int y;
   private:
      int z;
};

class A: Base
{
   // x is public
   // y is protected
   // z is not accessible from A
};

class B: protected Base
{
   // x is protected
   // y is protected
   // z is not accessible from B
};

class C : private Base
{
   // x is private
   // y is private
   // z is not accessible from C
};
class D: public Base
{
   // x is public
```

```
    // y is protected
    // z is not accessible from D
};
```

Question Two

Quicksort is an efficient sorting algorithm developed by British computer scientist Tony Hoare in 1959. One of its implementations is summarised in the pseudo code below with algorithm “quicksort” and algorithm “partition”:

```
algorithm quicksort(A, lo, hi) is
   if lo < hi then
        p := partition(A, lo, hi)
        quicksort(A, lo, p - 1)
        quicksort(A, p + 1, hi)

algorithm partition(A, lo, hi) is
    pivot := A[hi]
    i := lo
    for j := lo to hi do
        if A[j] < pivot then
            swap A[i] with A[j]
            i := i + 1
    swap A[i] with A[hi]
    return i
```

The algorithm “quicksort” divides a list of elements into two parts (called sub-lists) and perform sorting over the sub-lists recursively.

The algorithm “partition” selects a pivot to partition the list into sub-lists such that all elements in one sub-list are smaller than pivot and all elements in the other are greater than the pivot.

A) Given a list of {51,95,66,72,42,38,62}; illustrate the sorting process specified in the pseudo code
(10 marks)

B) Implement the pseudo code in C#

Model Answer:

A)

The 1st instance of the quick sort will partition the list as:

Sub-list {51, 42, 38}   Pivot: 62   and Sub-list {66, 72, 95}

The 2nd instance of the quick sort will be applied to {51, 42, 38} as sub-list {}, Pivot:38 and sub-list {51,42}

The 3rd instance of the quick sort will be applied to {42, 51} as Pivot:51 and sub-list {42}

The 4th instance of the quick sort will be applied to {66, 72, 95} as sub-list {66, 72}, Pivot:95

And then original list will be reassembled as:

{38,42,51,62,66,72,95}

B) The code below is for illustration purpose:

```
void Quick_Sort(int[] arr, int low , int high)
        {
            if (low < high)
            {
                int pivot = Partition(arr, low, high);

                if (pivot > low)
                {
                    Quick_Sort(arr, low, pivot - 1);
                }

                if (pivot < high)
                {
                    Quick_Sort(arr, pivot + 1, high);
                }

            }

        }


int Partition(int[] arr, int low, int high)
        {
            int pivot = arr[high];
            int i = low;  //place for swapping

            for (int j = low; j<=high-1; j++)
            {
                if (arr[j] <= pivot)
                {
                    Swap(arr, i, j);
                    i++;
                }
            }

 void Swap(int[] arr, int i, int j)
        {
            if (i!=j)
            {
                int tmp = arr[i];
                arr[i] = arr[j];
                arr[j] = tmp;
            }
        }
```

Question Three

In the context of Object-Oriented (OO) Analysis, there is a class-based modelling inheritance hierarchy which includes a super-class named as “Furniture” and a number of subclasses named as “Table”, “Chair”, “Desk”, and “Sofa”.

A) What is the meaning of “inheritance” in the context of OO class-based modelling?

Model Answer: It means that each subclass inherits all of the attributes and operations from its super-class, so all data structures and algorithms originally designed for the super-class are immediately available for its subclasses.

B) Please list five common attributes that the “Furniture” class may have and briefly discuss if these attributes can be used by its subclasses.

Model Answer: The common attributes of the “Furniture” class can be Make, Price, Model, Color, Materials etc. As they are the attributes of the super class so each subclass will inherit them and can use them.

C) Please briefly discuss the specific attributes that each subclass can have.

Model Answer: “Table” class can have specific features like Shape and Legs etc. “Chair” class have specific features like Height and Moveable; “Desk” class can have specific features like Shape and Drawers etc. “Sofa” class can have specific features like Sittings and Length etc.

Question Four

Unit testing and integration testing are two testing strategies and regression testing is a kind of integration testing. What does a unit testing do? and what does a regression testing do? Give a brief discussion and indicate why it is important to have integration testing.

Model Answer:

Unit testing is designed and used to test appropriate data and operations within a Unit, i.e. a class or component, to exercise all states of the unit

When changes (i.e. man-made errors or correction of errors) are made to an existing program, regression testing needs to be run to check whether these changes are propagated to other modules with unexpected side effects

The goal of integration testing is to make sure that independent modules that work correctly on their own do does not interfere with one another when they are integrated together since unforeseen side effects are always possible [8 marks].

SECTION B: ANSWER ALL QUESTIONS IN THIS SECTION

Question Five

What is a scalable system? Propose three general solutions to improve the scalability of a distributed software system. Provide one real-world example of a scalable software solution.

Model Answer:

A system whose performance improves after adding hardware, proportionally to the capacity added, is said to be a scalable system

Three typical solutions are:

Hiding communication latencies by modify the client side of the content management system. Try to avoid waiting for responses to remote (and potentially distant) service requests as much as possible. It implies use of asynchronous communication to construct the requesting application.

Distribute the application further on server side.

Distribution takes a component, splitting it into smaller parts, and subsequently spreads those parts across the system. It means to add more servers to separate the load. If done properly, this can drastically reduce the load any single server receives. However, this comes at the cost of added complexity.

More replication of application instance on server side.

Replication copies components across a distributed system. This is where it may make sense to scale your app server horizontally - basically, making copies of itself to split the load up between them.

Any cloud-based system (email, drop box, BREO etc), or software solutions which scale well.

Question Six – Mini Case Study

You have been assigned as project manager to develop a small, interactive system for a car hire company. The system will provide functionality for adding new members (clients), updating members' details, hiring cars and returning cars.

Discussions with the company staff have revealed the following facts: there are some confusion and uncertainty about the content and layout of the interface and also the requirements for adding new cars and updating cars details; the company's business line may soon be expanded from cars only to hire small vans as well.

Furthermore, discussions with your software engineers have revealed the following facts: the uncertainty about the requirements means that the total development time will be difficult to predict; some of the new team members are unfamiliar with testing tools; some key team members may be unavailable at critical times; the team are planning to use some reusable software components which to date have not been tested.

A) List **FIVE** different types of risk.

Model Answer:

Risk type
Technology
People
Organizational
Tools
Requirements
Estimation

B) Identify **ONE potential risk**, from the scenario above, for each type of risk that you have listed in part a).

Model Answer:

Risk Type	*Possible Risks*
Technology	*Reusable software may contain defects that mean they cannot be reused as planned.*
People	*Key development staff may be unavailable at critical times.*
Organizational	*A restructuring results in different management responsible for the project.*
Tools	*New team members are unfamiliar with testing tools.*
Requirements	*Client is unsure of some of the interface requirements.*
Estimation	*Development time is difficult to predict.*

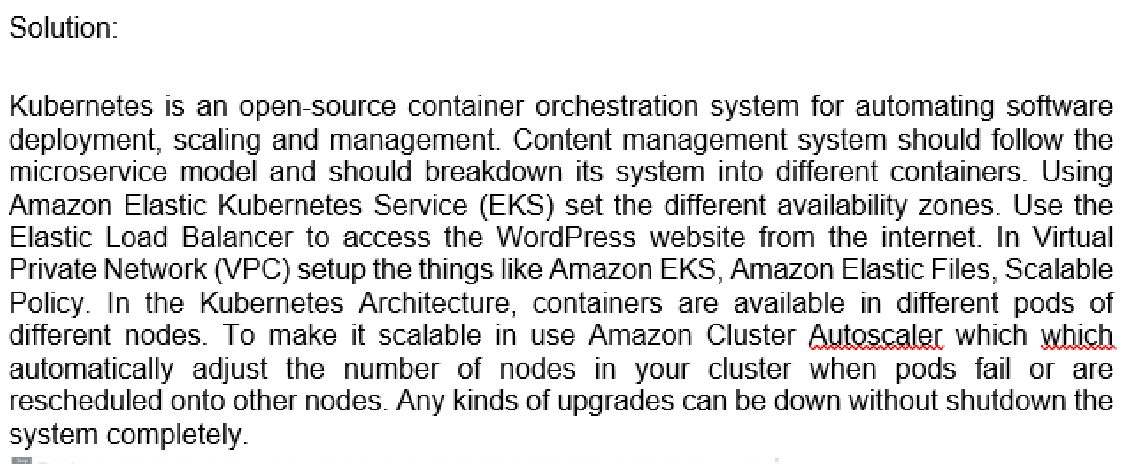
C) Develop a strategy for managing **FIVE** of the risks identified in part B). Your answer should be in the form of a table with ONE strategy for each risk.

Note: You do not need to re-write the Possible Risk in the table, numbers will be referenced to your answers in part b.

Model Answer:

Risk	Strategy

Reusable software may contain defects that mean they cannot be reused as planned.	Replace potentially defective components with bought-in components of known reliability.
Key development staff may be unavailable at critical times.	Reorganize teams so that there is more overlap of work and people
A restructuring results in different client management are responsible for the project.	Schedule meetings with new management team to explain current project requirements, stakeholders, risks, development
New team members are unfamiliar with testing tools.	Provide training to bring new members up to speed.
Client is unsure of some of the interface requirements.	Adopt paper prototyping approach to guide client through requirement elicitation.
Development time is difficult to predict.	Adopt an agile development approach such as SCRUM



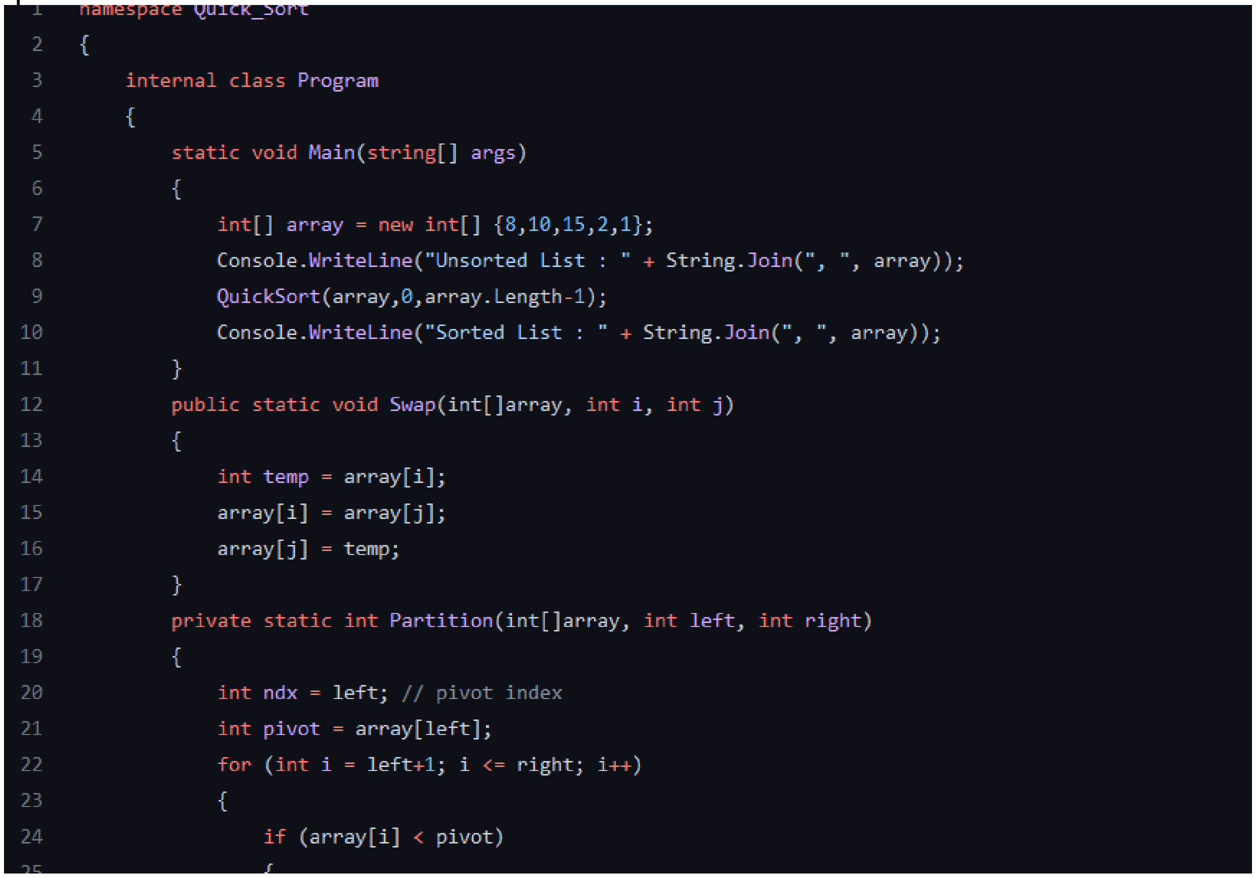
Solution:

Kubernetes is an open-source container orchestration system for automating software deployment, scaling and management. Content management system should follow the microservice model and should breakdown its system into different containers. Using Amazon Elastic Kubernetes Service (EKS) set the different availability zones. Use the Elastic Load Balancer to access the WordPress website from the internet. In Virtual Private Network (VPC) setup the things like Amazon EKS, Amazon Elastic Files, Scalable Policy. In the Kubernetes Architecture, containers are available in different pods of different nodes. To make it scalable in use Amazon Cluster Autoscaler which which automatically adjust the number of nodes in your cluster when pods fail or are rescheduled onto other nodes. Any kinds of upgrades can be down without shutdown the system completely.

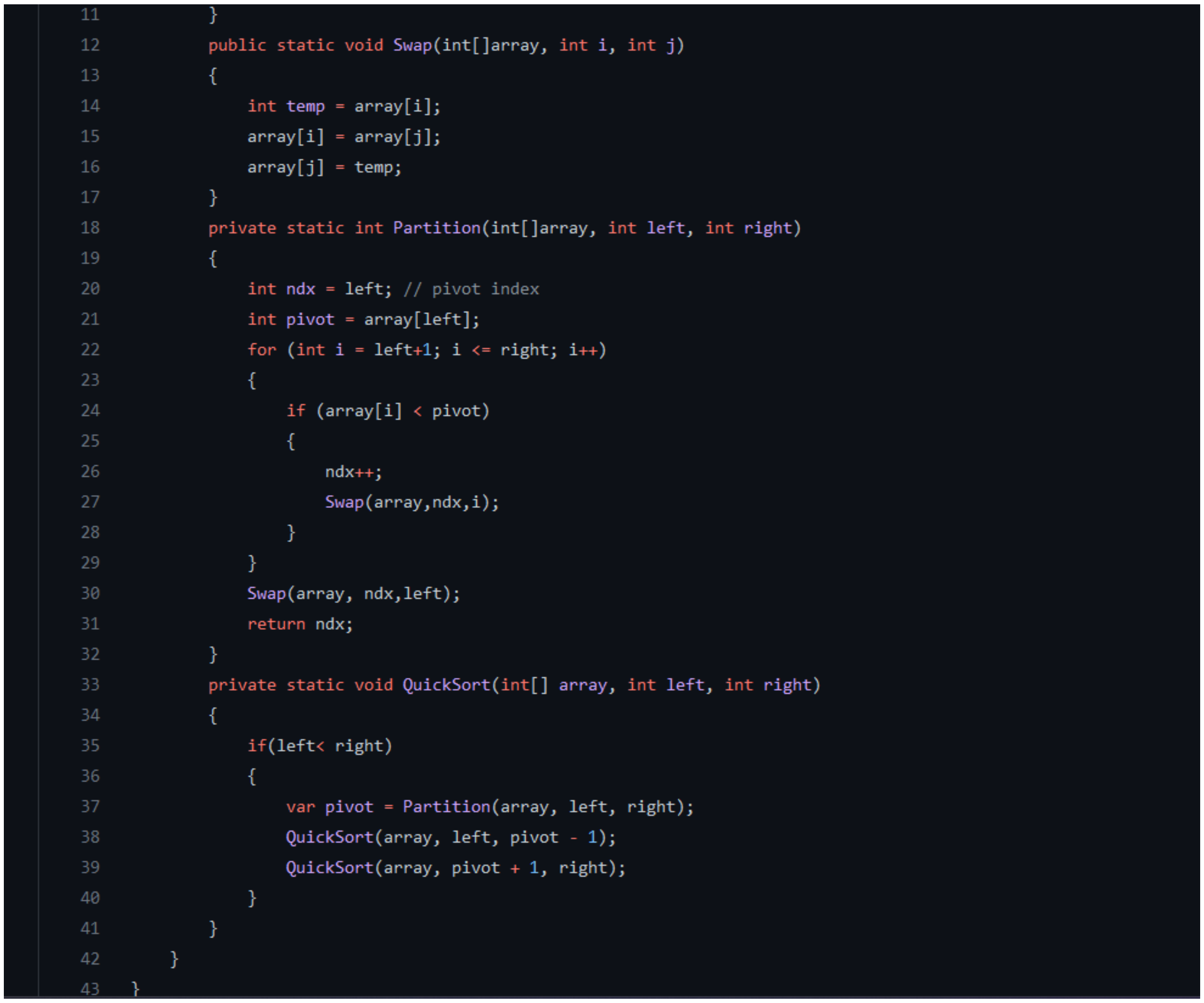
Cloud ans

content management system using WordPress has been set up on an Amazon Elastic Kubernetes Service (EKS) instance. After a few weeks, you realise that the system is starting to reach its limits and needs to be scaled up as soon as possible. To carry out the upgrade, the content management system has to be shut down for a period of time. Although, this is NOT a good practise for businesses. Continuous service delivery and continuous system integration are preferred. Propose a cloud-based solution which avoids the disruption such as shutdown completely? Elaborate on your cloud-based solution, its corresponding architecture and operation in detail

quick sort



```
namespace Quick_Sort
{
    internal class Program
    {
        static void Main(string[] args)
        {
            int[] array = new int[] {8,10,15,2,1};
            Console.WriteLine("Unsorted List : " + String.Join(", ", array));
            QuickSort(array,0,array.Length-1);
            Console.WriteLine("Sorted List : " + String.Join(", ", array));
        }
        public static void Swap(int[]array, int i, int j)
        {
            int temp = array[i];
            array[i] = array[j];
            array[j] = temp;
        }
        private static int Partition(int[]array, int left, int right)
        {
            int ndx = left; // pivot index
            int pivot = array[left];
            for (int i = left+1; i <= right; i++)
            {
                if (array[i] < pivot)
                {
```



```
        }
        public static void Swap(int[]array, int i, int j)
        {
            int temp = array[i];
            array[i] = array[j];
            array[j] = temp;
        }
        private static int Partition(int[]array, int left, int right)
        {
            int ndx = left; // pivot index
            int pivot = array[left];
            for (int i = left+1; i <= right; i++)
            {
                if (array[i] < pivot)
                {
                    ndx++;
                    Swap(array,ndx,i);
                }
            }
            Swap(array, ndx,left);
            return ndx;
        }
        private static void QuickSort(int[] array, int left, int right)
        {
            if(left< right)
            {
                var pivot = Partition(array, left, right);
                QuickSort(array, left, pivot - 1);
                QuickSort(array, pivot + 1, right);
            }
        }
    }
}
```

```
namespace Quick_Sort
{
internal class Program
{
static void Main(string[] args)
{
```

```
int[] array = new int[] {8,10,15,2,1};
Console.WriteLine("Unsorted List : " + String.Join(", ", array));
QuickSort(array,0,array.Length-1);
Console.WriteLine("Sorted List : " + String.Join(", ", array));
}
public static void Swap(int[]array, int i, int j)
{
int temp = array[i];
array[i] = array[j];
array[j] = temp;
}
private static int Partition(int[]array, int left, int right)
{
int ndx = left; // pivot index
int pivot = array[left];
for (int i = left+1; i <= right; i++)
{
if (array[i] < pivot)
{
ndx++;
Swap(array,ndx,i);
}
}
Swap(array, ndx,left);
return ndx;
}
private static void QuickSort(int[] array, int left, int right)
{
if(left< right)
{
```

```
var pivot = Partition(array, left, right);
QuickSort(array, left, pivot - 1);
QuickSort(array, pivot + 1, right);
}
}
}
}
```

Bubble sort

```
namespace Bubble_Sort
{
internal class Program
{
static void Main(string[] args)
{
double[] unsortedList = { 36, 2, 29, 1, 8, 14};
Console.WriteLine("Unsorted List : " + String.Join(",", unsortedList));
double[] sortedList = BubbleSort(unsortedList);
Console.WriteLine("Sorted List : " + String.Join(",",sortedList));

}
public static double[] BubbleSort(double[] unsortedList)
{
double temp;
for (int i = 0; i < unsortedList.Length-1; i++)
{
for(int j = 0; j < unsortedList.Length -(1+i); j++)
{
if(unsortedList[j] > unsortedList[j + 1])
{
temp = unsortedList[j+1];
unsortedList[j+1] = unsortedList[j];
unsortedList[j] = temp;
}
}
}
return unsortedList;
}
}
}
```

Insectrion sort

```
namespace Insertion_Sort
{
internal class Program
{
static void Main(string[] args)
{
double[] unsortedListBest = { 4, 6, 7, 8, 10};
double[] unsortedListAverage = { 10, 4, 8, 6, 7};
double[] unsortedListWorst = { 10, 8, 7, 6, 4 };

Console.WriteLine("Unsorted List : " + String.Join(", ", unsortedListWorst));
double[] sortedList = InsertionSort(unsortedListWorst);
Console.WriteLine("Sorted List : " + String.Join(",", sortedList));

}
private static double[] InsertionSort(double[] unsortedList)
{
int i = 1;
int j = i;
double temp = 0;
while(i < unsortedList.Length)
{
j = i;
while(j > 0 && unsortedList[j -1] > unsortedList[j])
{
temp = unsortedList[j];
unsortedList[j] = unsortedList[j - 1];
unsortedList[j - 1] = temp;
j--;
}
i++;
}
return unsortedList;
}
}
}
```

Selection sort

```
using System.Linq;

namespace Selection_Sort
{
internal class Program
{
static void Main(string[] args)
{
double[] unsortedList = { 10,4,8,6,7};
Console.WriteLine("Unsorted List :" + String.Join(", ", unsortedList));
double[] sortedList = SelectionSort(unsortedList);
Console.WriteLine("Sorted List : " + String.Join(", ",sortedList));

}
private static double[] SelectionSort(double[] unsortedList)
{
int Minimum_Index = 0;
double Minimum_Value_Found = 0;
for (int MainIndex =0; MainIndex<unsortedList.Length;MainIndex++)
{
Minimum_Index = MainIndex;
for(int Remaining_Index = MainIndex + 1; Remaining_Index < unsortedList.Length;
Remaining_Index++)
{
if(unsortedList[Remaining_Index] < unsortedList[Minimum_Index])
{
Minimum_Index = Remaining_Index;
}
}
Minimum_Value_Found = unsortedList[Minimum_Index];
unsortedList[Minimum_Index] = unsortedList[MainIndex];
unsortedList[MainIndex] = Minimum_Value_Found;
}
return unsortedList;

}
}
}
```

What is Agile? Agile is an iterative approach to project management and software development that helps teams deliver value to their customers faster and with fewer headaches. Instead of betting everything on a "big bang" launch, an agile team delivers work in small, but consumable, increments. Requirements, plans, and results are evaluated continuously so teams have a natural mechanism for responding to change quickly. gile processes generally promote a disciplined project management process that encourages frequent inspection and adaptation, a leadership philosophy that encourages teamwork, self-organization and accountability, a set of engineering best practices intended to allow for rapid delivery of high-quality software, and a business approach that aligns development with customer needs and company goals.

Components of Agile : - Requirements gathering Design the requirements Construction/ iteration Testing/ Quality assurance Deployment Feedback

why is it called Scrum? People often ask, “Is Scrum an acronym for something?” and the answer is no. It is actually inspired by a scrum in the sport of rugby. In rugby, the team comes together in what they call a scrum to work together to move the ball forward. In this context, Scrum is where the team comes together to move the product forward.

What is Scrum? People often ask, “Is Scrum an acronym for something?” and the answer is no. It is actually inspired by a scrum in the sport of rugby. In rugby, the team comes together in what they call a scrum to work together to move the ball forward. In this context, Scrum is where the team comes together to move the product forward. Scrum is an empirical process, where decisions are based on observation, experience and experimentation. Scrum has three pillars: transparency, inspection and adaptation.

Scrum is an empirical process, where decisions are based on observation, experience and experimentation. Scrum has three pillars: transparency, inspection and adaptation.

Scrum Events Events that create regularity and minimize other meetings Sprint - short cycles of one month or less, during which the work is done; the Sprint contains all of the other Scrum events; a new Sprint starts immediately after the conclusion of the previous Sprint Sprint Planning - event dedicated to planning out the work that will take place during the Sprint Daily Scrum - event held every day where the Developers inspect the progress toward the Sprint Goal, uncover anything that may be getting in their way and adapt accordingly Sprint Review - event held at the end of the Sprint where the Scrum Team and key stakeholders review what was accomplished in the Sprint and what has changed in their environment; next, attendees collaborate on what to do next Sprint Retrospective - the Scrum Team gets together during this event to talk about how the last Sprint went and identify the most helpful changes to improve their effectiveness