## 1. Difference between Agile Testing and Scrum.

Answer:-

**Agile testing** :- Agile testing is a type of software testing that is performed alongside the software development. It is the part of Agile Software Development in which the development and the testing process goes side by side. When in software development life cycle development and the testing goes parallel way then it is known as agile development process.
**Scrum testing** :- Scrum testing is a type of software testing that is the part of agile testing. It is basically a framework used in agile software development process. Scrum testing is used to design the complicated software. It also checks the quality, performance and the usability of the software.

### Agile

1. Agile is an iterative and incremental approach to software development methodology.
2. In this approach, the leadership plays an important role.
3. Agile software development is highly suitable for the medium or large project.
4. Flexibility is the most significant advantage of agile as it quickly reacts to changes.
5. Agile involves face-to-face communication and collaboration between the members of various cross-functional teams.
6. Agile development needs frequent delivery to the end user for their feedback.
7. In this development, each step like requirements, analysis, design, are continually monitored during the lifecycle.
8. The project leader takes care of all the tasks in the agile method.

## Scrum

1. Scrum is a framework of agile methodology. In which incremental builds are delivered to end user in every two to three weeks.
2. Scrum's team is self-organized, cross-functional team.
3. Scrum is used in the project where the requirement rapidly changes. v
4. A compared to agile it is more rigid. So that there are no chances of frequent change.
5. In daily stand up meeting the teamwork is achieved with a fixed role assigned to team members, scrum master, and product owner.
6. No need to change many more while implementing scrum process.
7. In this process, a build is delivered after each sprint to the client for their feedback.
8. After every sprint a demonstration of functionality is provided. So that the regular feedback can be taken before next sprint.

## KEY DIFFERENCE

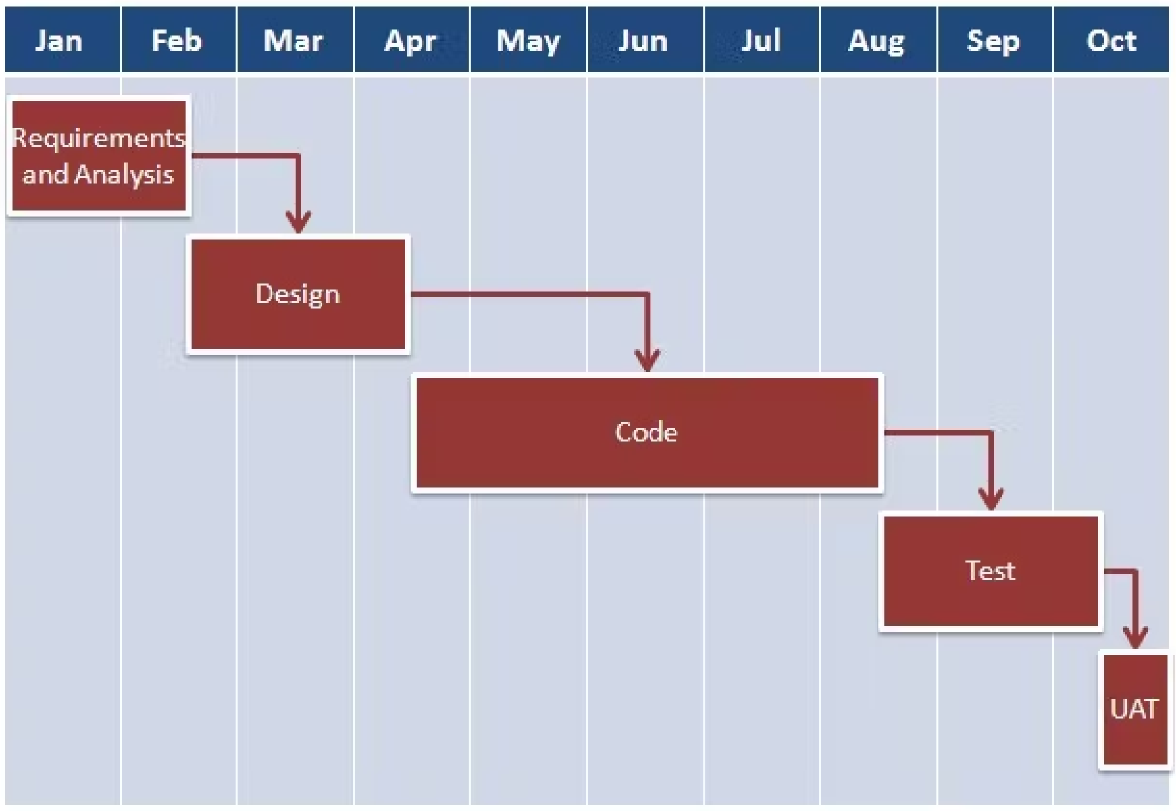
1. Agile is a continuous iteration of development and testing in the software development process whereas Scrum is an Agile process to focus on delivering the business value in the shortest time.
2. Agile methodology delivers the software on a regular basis for feedback while Scrum delivers the software after each sprint.
3. In the Agile process, leadership plays a vital role; on the other hand, Scrum fosters a self-organizing, cross-functional team.
4. Agile involves collaborations and face-to-face interactions between the members of various cross-functional teams whereas Scrum collaboration is achieved in daily stand up meetings.
5. In Agile process design and execution should be kept simple whereas in Scrum process design and execution can be innovative and experimental.

**2 .Adobe is working on a project to come up with a competing product for Microsoft Word, that provides all the features provided by Microsoft Word and any other features requested by the marketing team. The final product needs to be ready in 10 months of time. Explain this project flow using Agile Methodologies.**

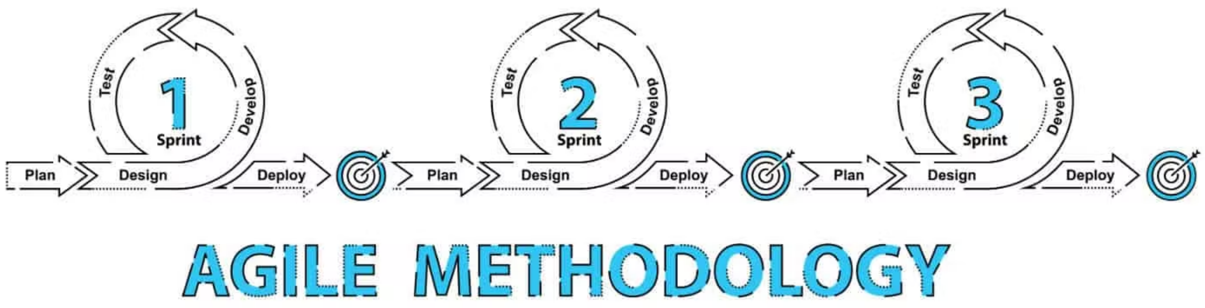
Answer:-

## 2.1 In traditional Waterfall model

1. At a high level, the project teams would spend 15% of their time on gathering requirements and analysis (1.5 months)
2. 20% of their time on design (2 months)
3. 40% on coding (4 months) and unit testing
4. 20% on System and Integration testing (2 months).
5. At the end of this cycle, the project may also have 2 weeks of User Acceptance testing by marketing teams.
6. In this approach, the customer does not get to see the end product until the end of the project, when it becomes too late to make significant changes.

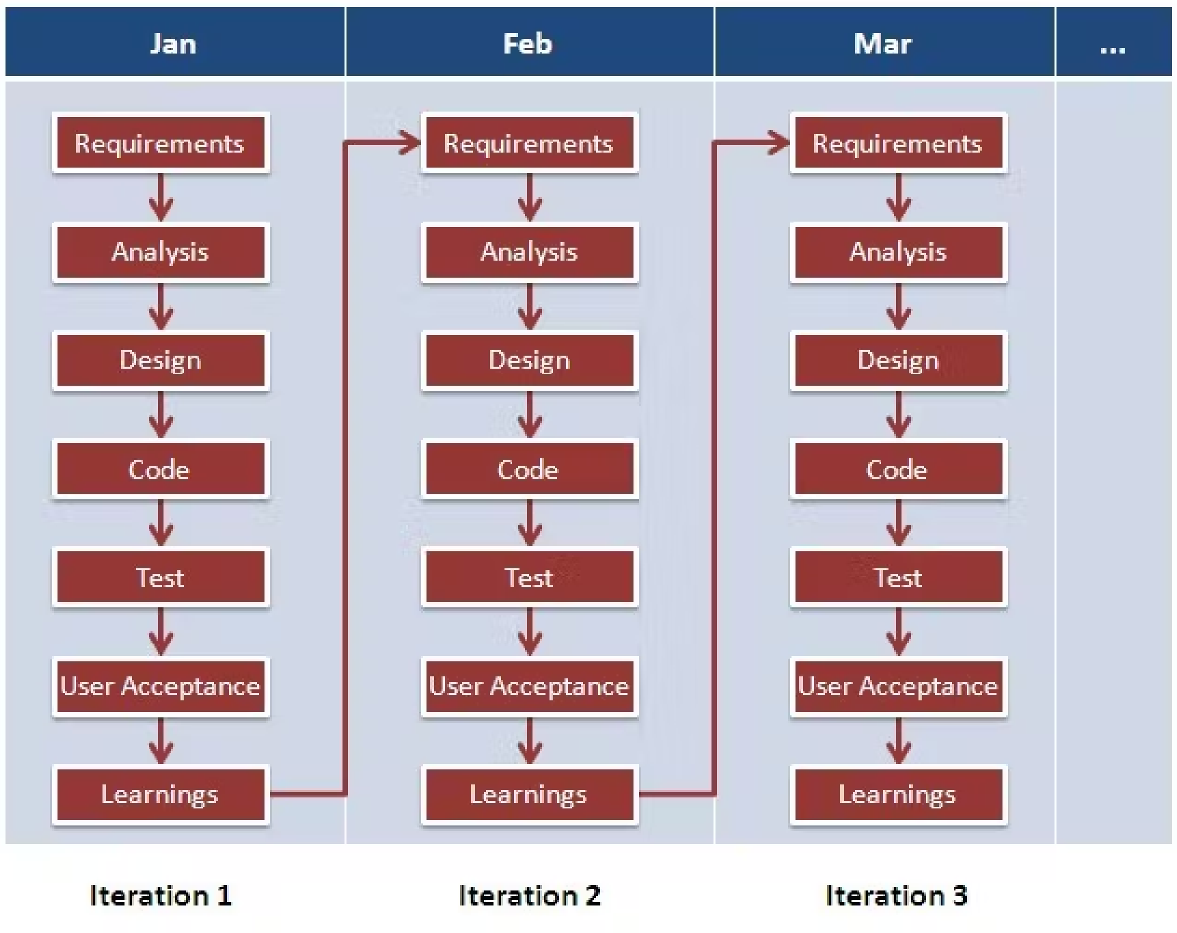


### 2.2 With Agile development methodology

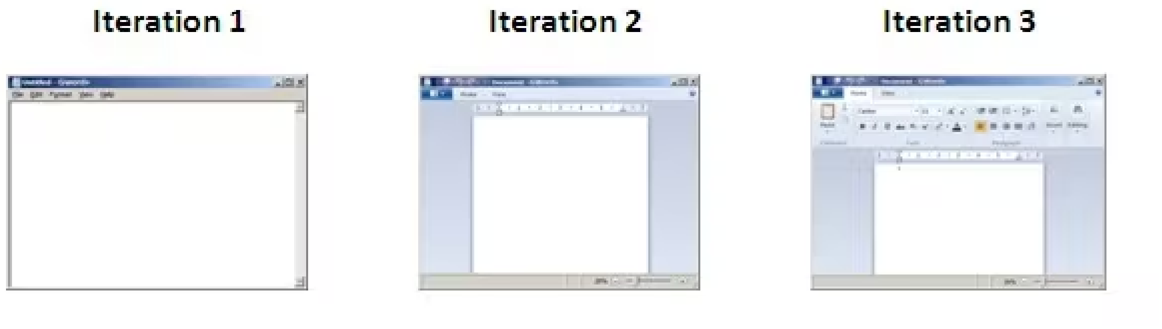


1. In the Agile methodology, each project is broken up into several ‘Iterations’.
2. All Iterations should be of the same time duration (between 2 to 8 weeks).
3. At the end of each iteration, a working product should be delivered.

4. In simple terms, in the Agile approach the project will be broken up into 10 releases (assuming each iteration is set to last 4 weeks).
5. Rather than spending 1.5 months on requirements gathering, in Agile software development, the team will decide the basic core features that are required in the product and decide which of these features can be developed in the first iteration.
6. Any remaining features that cannot be delivered in the first iteration will be taken up in the next iteration or subsequent iterations, based on priority.
7. At the end of the first iterations, the team will deliver a working software with the features that were finalized for that iteration.
8. There will be 10 iterations and at the end of each iteration the customer is delivered a working software that is incrementally enhanced and updated with the features that were shortlisted for that iteration.



This approach allows the customer to interact and work with functioning software at the end of each iteration and provide feedback on it. This approach allows teams to take up changes more easily and make course corrections if needed. In the Agile approach, software is developed and released incrementally in the iterations. An example of how software may evolve through iterations is shown in the image below.

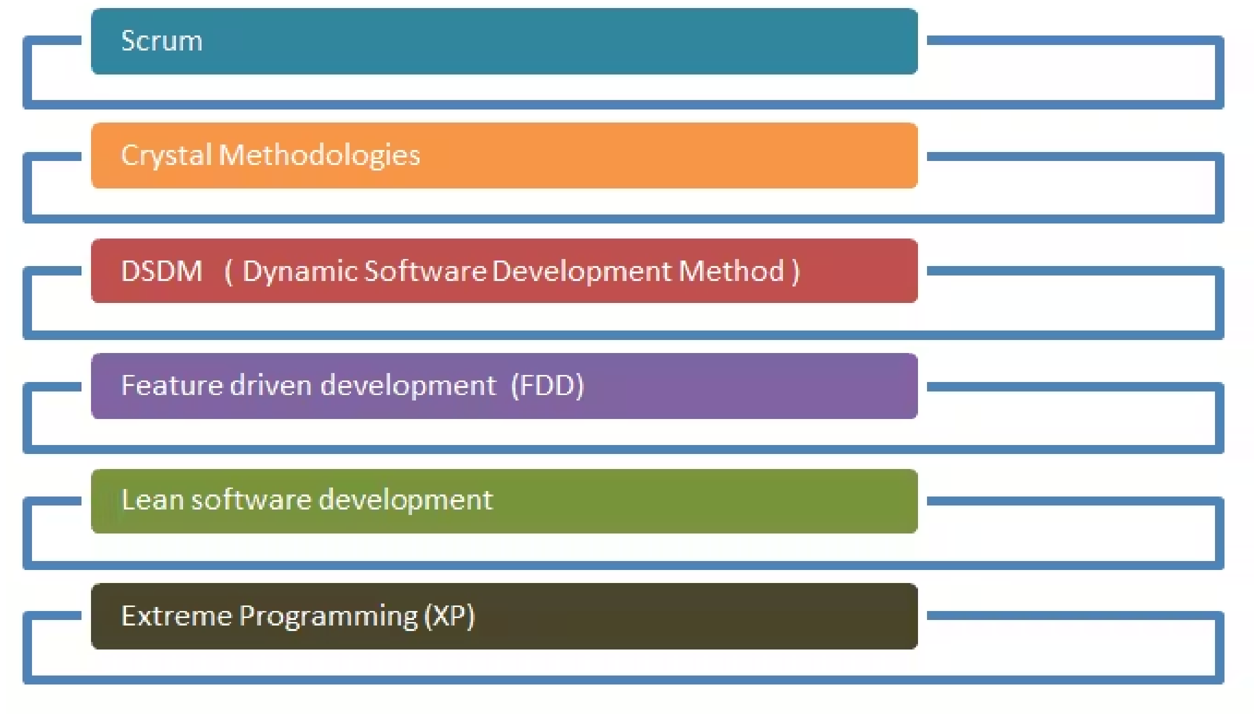


## 2.3 What is Agile Methodology ?

AGILE methodology is a practice that promotes continuous iteration of development and testing throughout the software development lifecycle of the project. In the Agile model, both development and testing activities are concurrent, unlike the Waterfall model.

### 2.3.1 Agile Process

Check below Agile process model to deliver successful systems quickly.

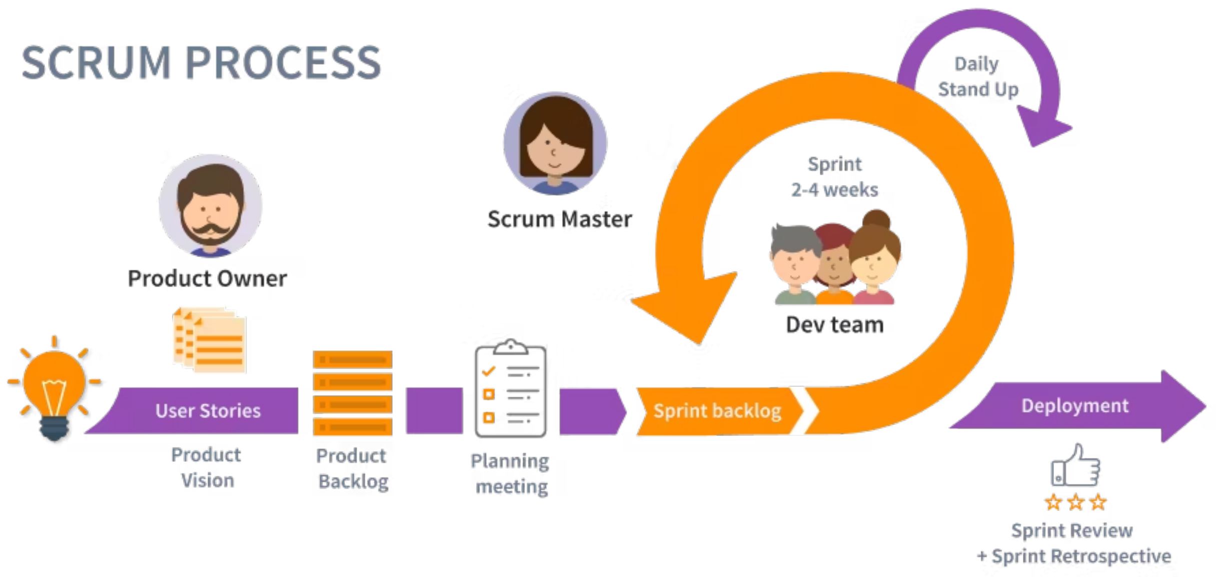

Scrum
Crystal Methodologies
DSDM ( Dynamic Software Development Method )
Feature driven development (FDD)
Lean software development
Extreme Programming (XP)

There are various Agile methods present in agile testing, and those are listed

### 2.3.1.a) Scrum

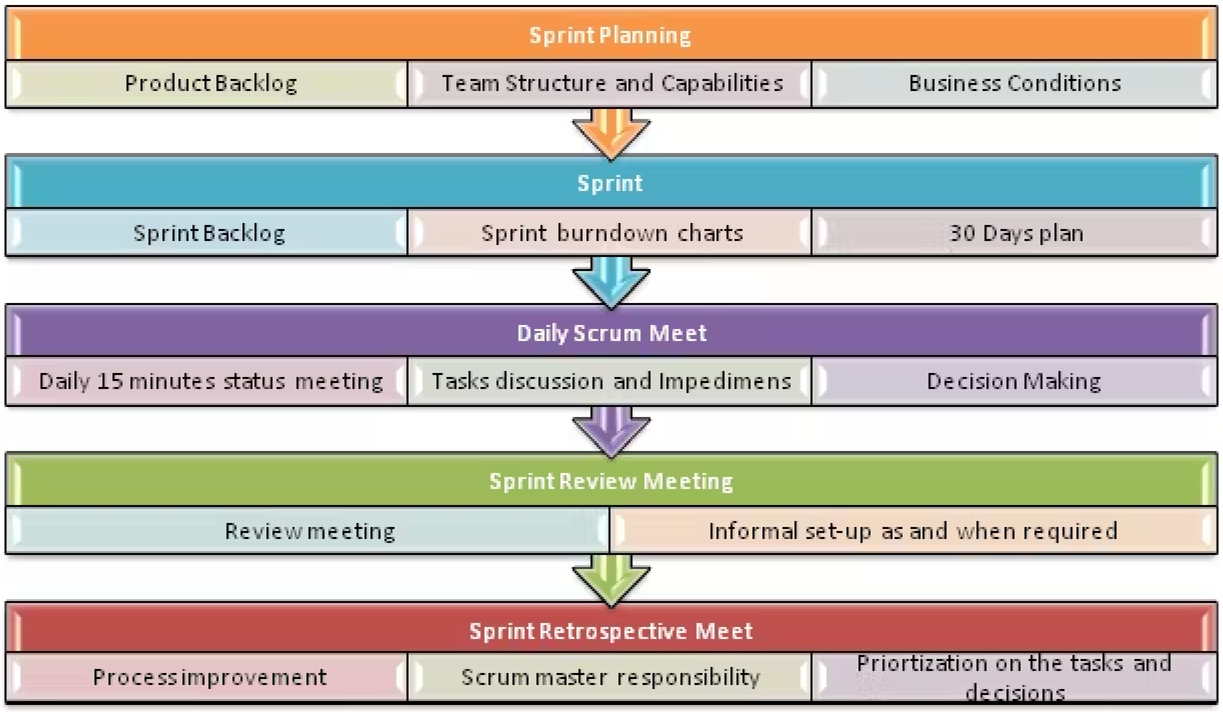
SCRUM is an agile development method which concentrates specifically on how to manage tasks within a team-based development environment. Basically, *Scrum is derived from activity that occurs during a rugby match*. Scrum believes in empowering the development team and advocates working in small teams (say- 7 to 9 members). It consists of three roles, and their responsibilities are explained as follows:

1. Scrum Master: Master is responsible for setting up the team, sprint meeting and removes obstacles to progress
2. Product owner: The Product Owner creates product backlog, prioritizes the backlog and is responsible for the delivery of the functionality at each iteration
3. Scrum Team: Team manages its own work and organizes the work to complete the sprint or cycle



Product Backlog This is a repository where requirements are tracked with details on the no of requirements(user stories) to be completed for each release. It should be maintained and prioritized by Product Owner, and it should be distributed to the scrum team. Team can also request for a new requirement addition or modification or deletion.

## Scrum Practices Practices are described in detailed:

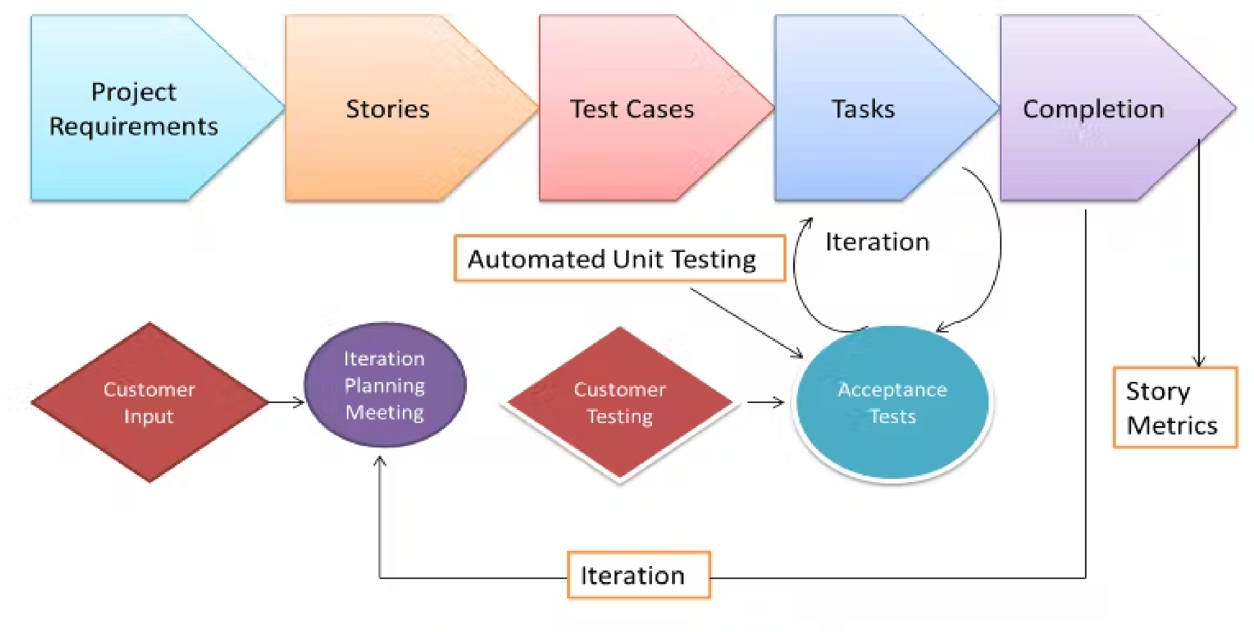

Sprint Planning
Product Backlog
Team Structure and Capabilities
Business Conditions
Sprint
Sprint Backlog
Sprint burndown charts
30 Days plan
Daily Scrum Meet
Daily 15 minutes status meeting
Tasks discussion and Impedimens
Decision Making
Sprint Review Meeting
Review meeting
Informal set-up as and when required
Sprint Retrospective Meet
Process improvement
Scrum master responsibility
Priortization on the tasks and decisions

Process flow of Scrum Methodologies Process flow of scrum testing is as follows:

1. Each iteration of a scrum is known as Sprint
2. Product backlog is a list where all details are entered to get the end-product
3. During each Sprint, top user stories of Product backlog are selected and turned into Sprint backlog
4. Team works on the defined sprint backlog
5. Team checks for the daily work
6. At the end of the sprint, team delivers product functionality.

### 2.3.1.b) eXtreme Programming (XP)

Extreme Programming technique is very helpful when there is constantly changing demands or requirements from the customers or when they are not sure about the functionality of the system. It advocates frequent "releases" of the product in short development cycles, which inherently improves the productivity of the system and also introduces a checkpoint where any customer requirements can be easily implemented. The XP develops software keeping customer in the target.



Extreme Programming (XP)

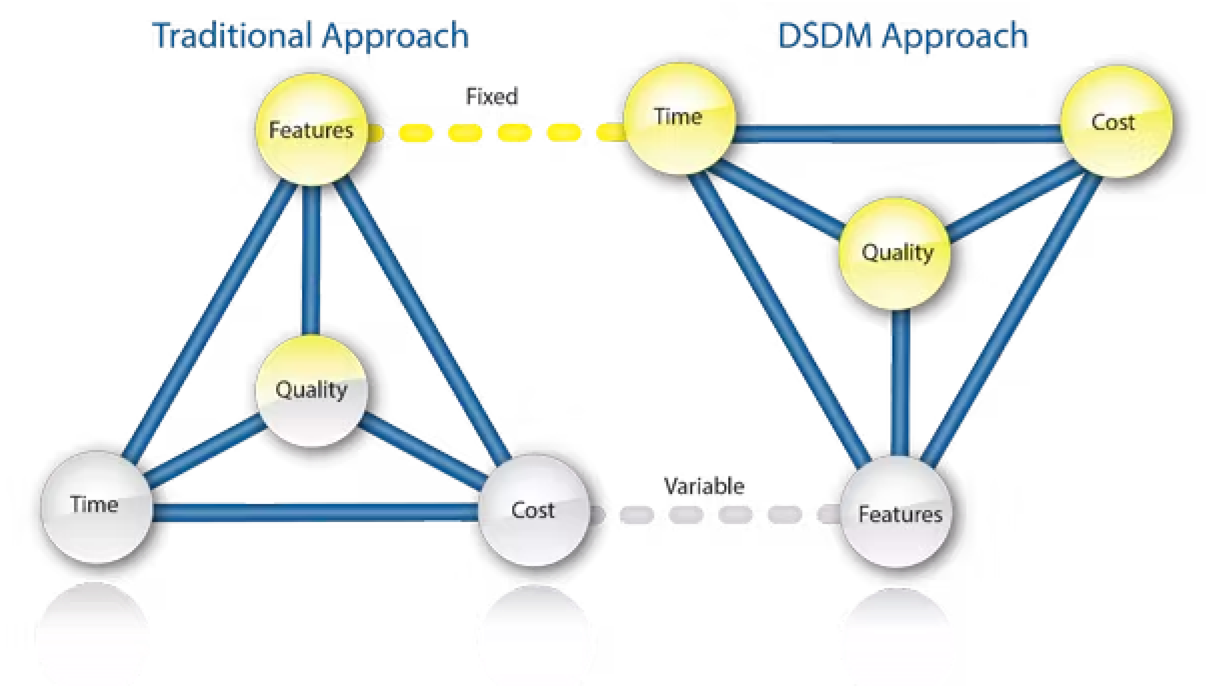
Business requirements are gathered in terms of stories. All those stories are stored in a place called the parking lot.

In this type of methodology, releases are based on the shorter cycles called Iterations with span of 14 days time period. Each iteration includes phases like coding, unit testing and system testing where at each phase some minor or major functionality will be built in the application.

### 2.3.1.b) Crystal Methodologies

Crystal Methodology is based on three concepts: (a) Chartering (b) Cyclic delivery (c) Wrap up

### 2.3.1.c) Dynamic Software Development Method (DSDM)



DSDM is a Rapid Application Development (RAD) approach to software development and provides an agile project delivery framework. The important aspect of DSDM is that the users are required to be involved actively, and the teams are given the power to make decisions. Frequent delivery of product becomes the active focus with DSDM.

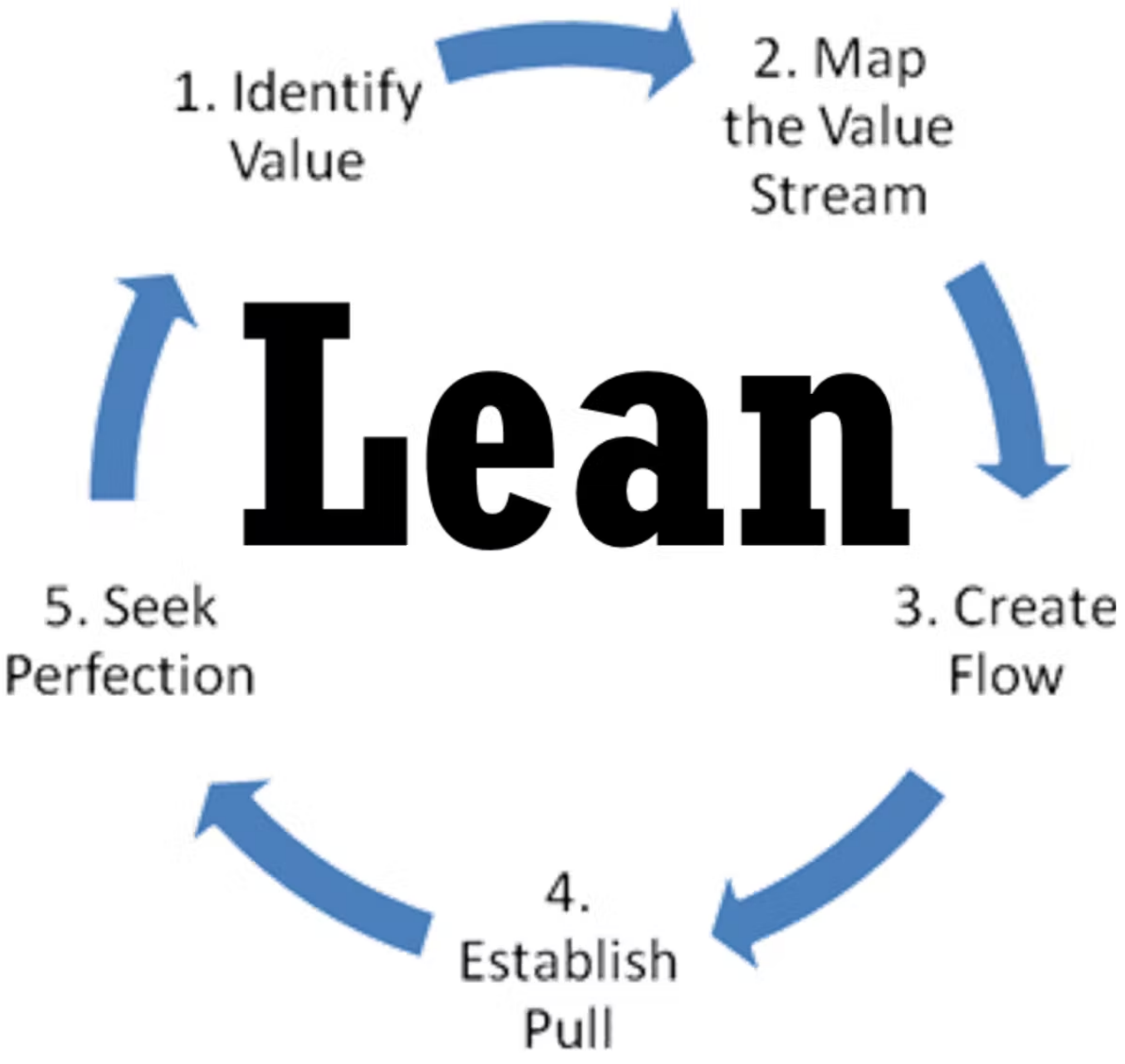
The techniques used in DSDM are: Time Boxing, MoSCoW Rules and Prototyping. The DSDM project consists of 7 phases: Pre-project, Feasibility Study, Business Study, Functional Model Iteration, Design and build Iteration, Implementation, Post-project.

MoSCoW (Must Have, Should Have, Could Have, Won't Have this time) is primarily used to prioritise requirements, although the practice is also useful in many other areas.

### 2.3.1.d) Feature Driven Development (FDD)

his method is focused around "designing & building" features. Unlike other agile methods, FDD describes very specific and short phases of work that has to be accomplished separately per feature. It includes domain walkthrough, design inspection, promote to build, code inspection and design.

### 2.3.1.e) Lean Software Development



Lean software development method is based on the principle "Just in time production". It aims at increasing speed of software development and decreasing cost. Lean development can be summarized in seven steps.

1. Eliminating Waste
2. Amplifying learning

3. Defer commitment (deciding as late as possible)
4. Early delivery
5. Empowering the team
6. Building Integrity
7. Optimize the whole

**3. What is kernel and its usage in UNIX? Explain any 8 commands of UNIX with examples?**

Answer:-

In computer science, Kernel is a computer program that is a core or heart of an operating system. Before discussing kernel in detail, let's first understand its basic, i.e., Operating system in a computer.

The Unix operating system is a set of programs that act as a link between the computer and the user.

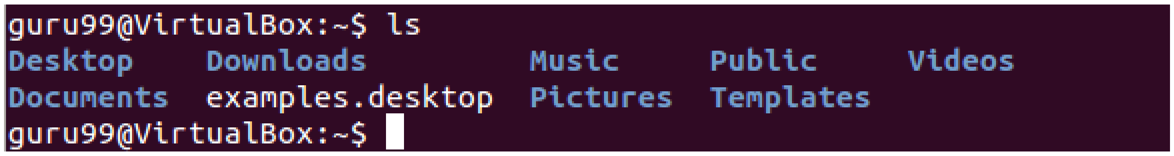
The computer programs that allocate the system resources and coordinate all the details of the computer's internals is called the operating system or the kernel.

Users communicate with the kernel through a program known as the shell. The shell is a command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel.

1. Unix was originally developed in 1969 by a group of AT&T employees Ken Thompson, Dennis Ritchie, Douglas McIlroy, and Joe Ossanna at Bell Labs.
2. There are various Unix variants available in the market. Solaris Unix, AIX, HP Unix and BSD are a few examples. Linux is also a flavor of Unix which is freely available.
3. Several people can use a Unix computer at the same time; hence Unix is called a multiuser system.

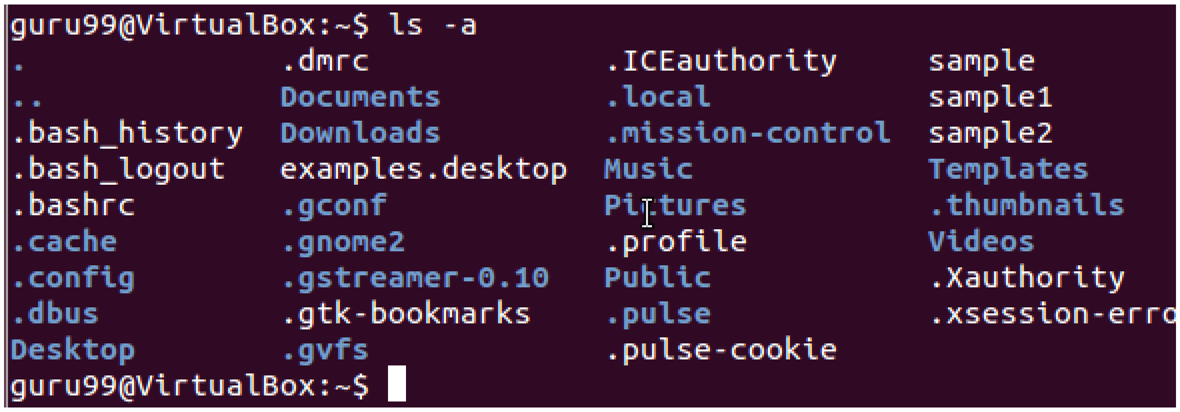
4. A user can also run multiple programs at the same time; hence Unix is a multitasking environment.

**1)Listing files (ls) :-** If you want to see the list of files on your UNIX or Linux system, use the 'ls' command.It shows the files /directories in your current directory.



```
guru99@VirtualBox:~$ ls
Desktop    Downloads         Music     Public     Videos
Documents  examples.desktop  Pictures  Templates
guru99@VirtualBox:~$ 
```

**2)Listing Hidden Files (`ls -a`):-** Hidden items in UNIX/Linux begin with – at the start of the file or directory. Any Directory/file starting with a '.' will not be seen unless you request for it. To view hidden files, use the command.



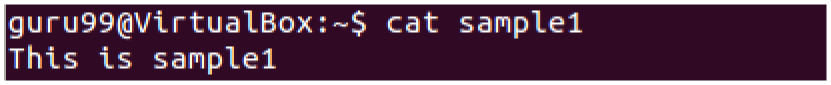
```
guru99@VirtualBox:~$ ls -a
.               .dmrc              .ICEauthority     sample
..              Documents          .local            sample1
.bash_history   Downloads          .mission-control  sample2
.bash_logout    examples.desktop   Music             Templates
.bashrc         .gconf             Pictures          .thumbnails
.cache          .gnome2            .profile          Videos
.config         .gstreamer-0.10    Public            .Xauthority
.dbus           .gtk-bookmarks     .pulse            .xsession-erro
Desktop         .gvfs              .pulse-cookie
guru99@VirtualBox:~$ 
```

**3)Creating()& Viewing Files(cat filename)**

To create a new file, use the command

1. cat > filename
2. Add content
3. Press 'ctrl + d' to return to command prompt.

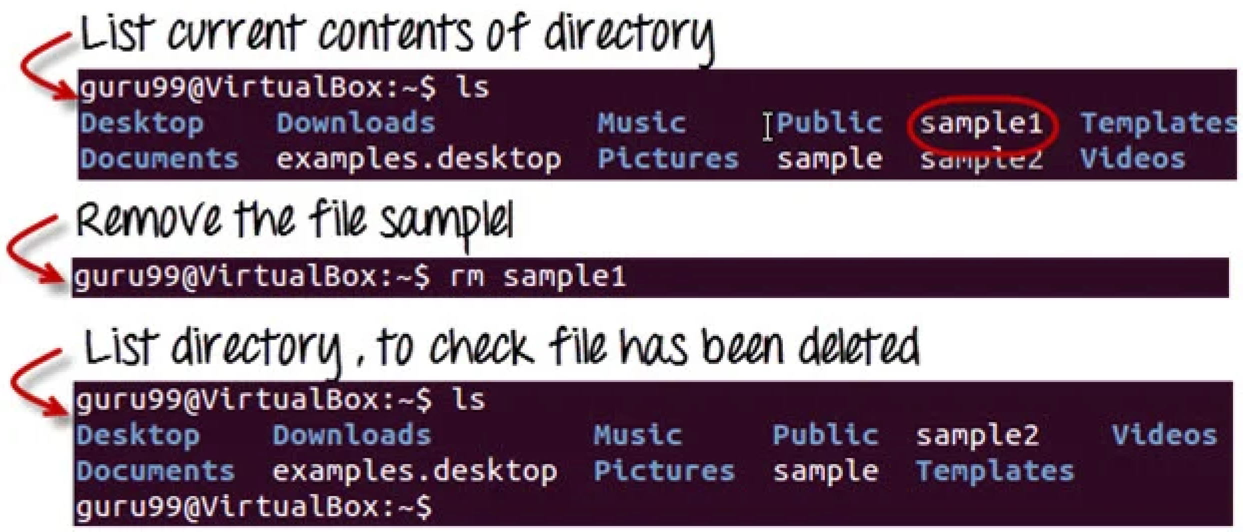
Let's see the file we just created –



```
guru99@VirtualBox:~$ cat sample1
This is sample1
```

**4) Deleting Files(`rm filename`)**

The ‘rm’ command removes files from the system without confirmation



List current contents of directory

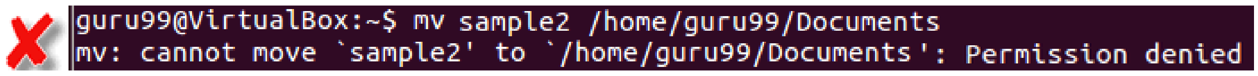
```
guru99@VirtualBox:~$ ls
Desktop     Downloads          Music     Public   sample1   Templates
Documents   examples.desktop   Pictures  sample   sample2   Videos
```

```
guru99@VirtualBox:~$ rm sample1
```

List directory , to check file has been deleted

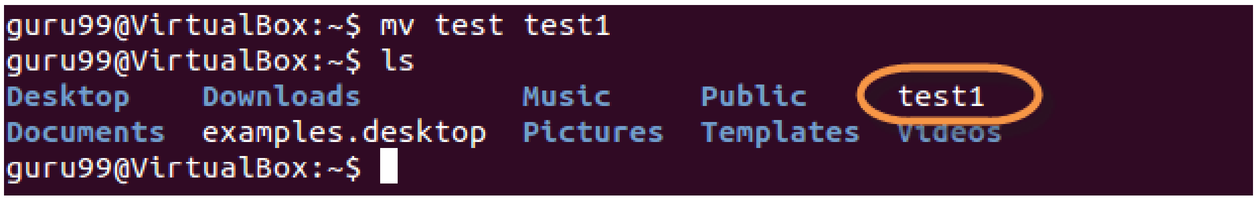
```
guru99@VirtualBox:~$ ls
Desktop     Downloads          Music     Public   sample2    Videos
Documents   examples.desktop   Pictures  sample   Templates
guru99@VirtualBox:~$
```

**5)Moving (`mv filename new_file_location`)**
To move a file, use the command.



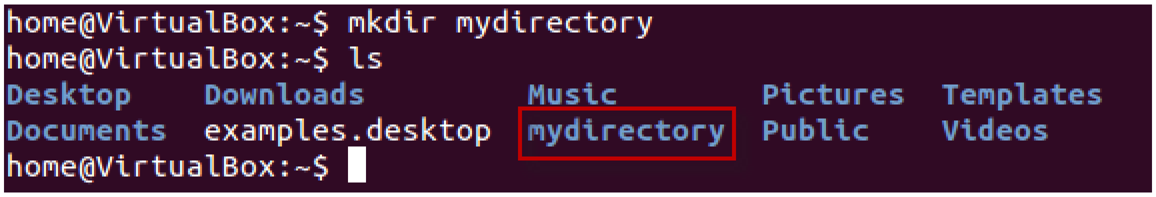
```
guru99@VirtualBox:~$ mv sample2 /home/guru99/Documents
mv: cannot move `sample2' to `/home/guru99/Documents': Permission denied
```

**6)For renaming file(`mv filename newfilename`):-**



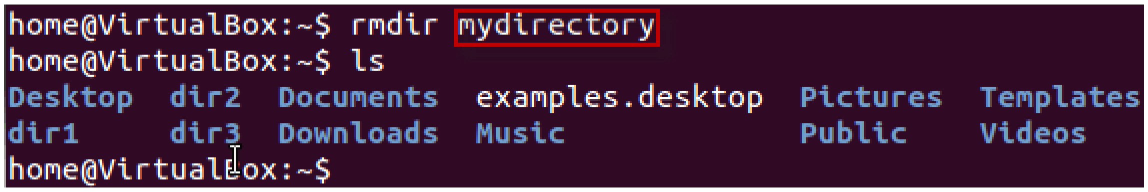
```
guru99@VirtualBox:~$ mv test test1
guru99@VirtualBox:~$ ls
Desktop     Downloads          Music     Public      test1
Documents   examples.desktop   Pictures  Templates   Videos
guru99@VirtualBox:~$
```

**7)Creating Directories(`mkdir mydirectory`):-** Directories can be created on a Linux operating system using the following command



```
home@VirtualBox:~$ mkdir mydirectory
home@VirtualBox:~$ ls
Desktop     Downloads         Music        Pictures  Templates
Documents   examples.desktop  mydirectory  Public    Videos
home@VirtualBox:~$
```

8)**Removing Directories( `rmdir mydirectory`):-** To remove a directory, use the command –



```
home@VirtualBox:~$ rmdir mydirectory
home@VirtualBox:~$ ls
Desktop  dir2  Documents  examples.desktop  Pictures  Templates
dir1     dir3  Downloads  Music             Public    Videos
home@VirtualBox:~$
```

**4. Mars' gravity is about 38% of Earth's. Write a program that will calculate your weight on Mars.**

● Declare all variables at the top of the class.

● Initial variables are to be of float type.

● After making the calculations, assign the result to a new variable, this time of the double type.

● After assigning the assignment, write the variable double to the console, limiting its length to 4 decimal places.

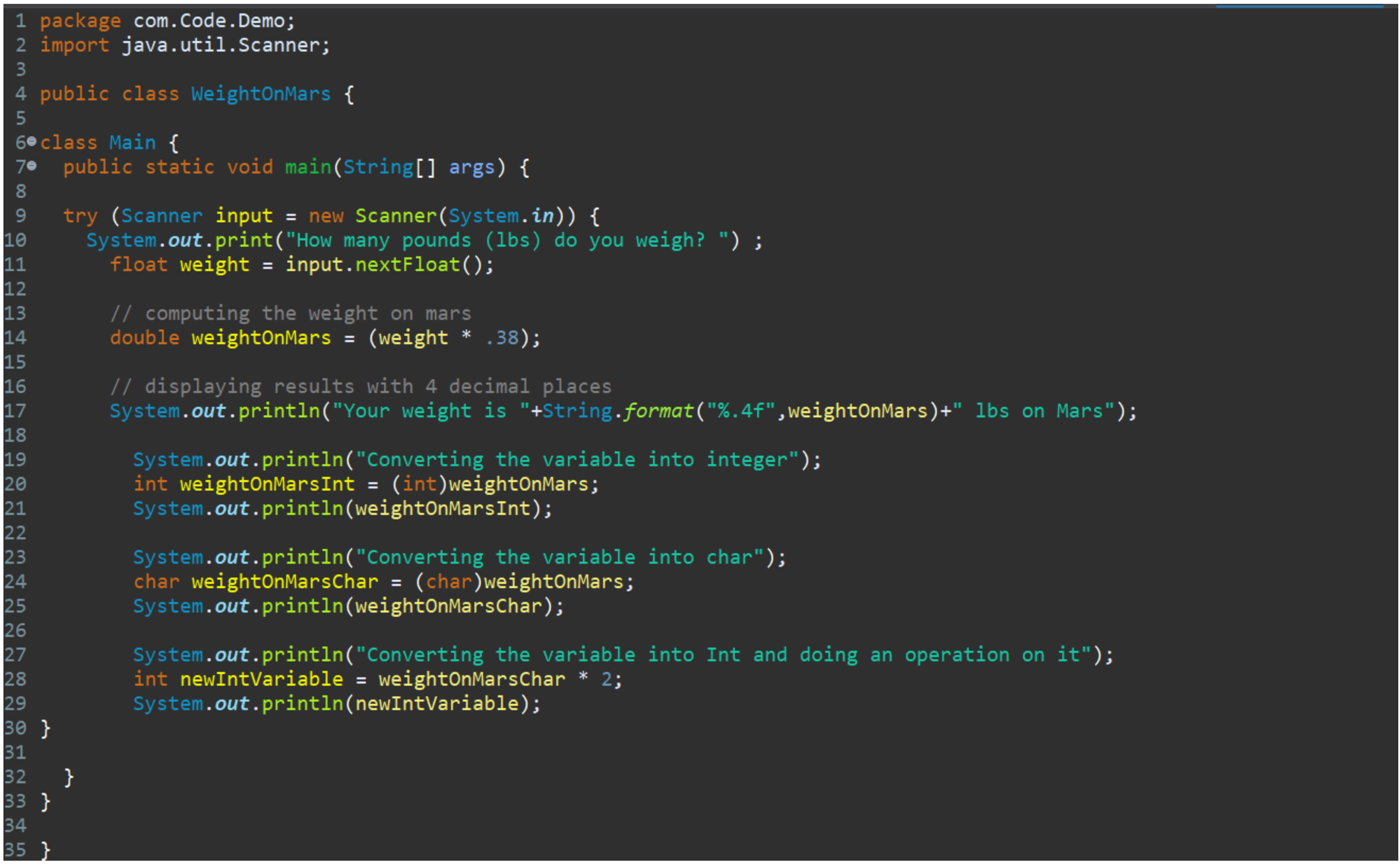
● Cast the above variable of double type to a new variable of int type.

● Cast the above variable of type int to a new variable of type char.

● Do any math operation on this variable char type and assign the value of this activity to the new variable int type.

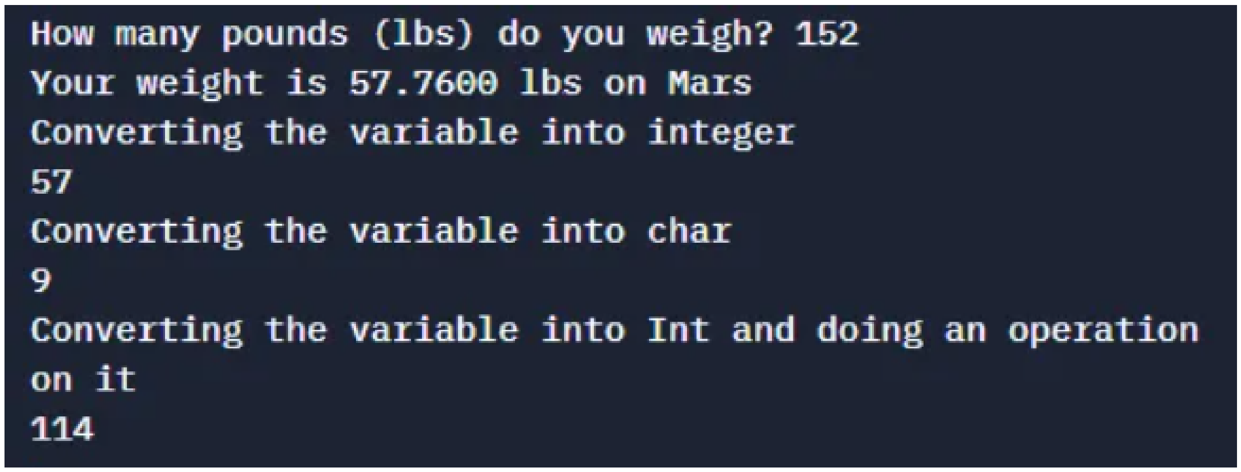
● Each of the above actions should be written to the console, adding some text explaining what has been done

**PROGRAM :-**



```java
package com.Code.Demo;
import java.util.Scanner;

public class WeightOnMars {

class Main {
  public static void main(String[] args) {

  try (Scanner input = new Scanner(System.in)) {
    System.out.print("How many pounds (lbs) do you weigh? ") ;
      float weight = input.nextFloat();

      // computing the weight on mars
      double weightOnMars = (weight * .38);

      // displaying results with 4 decimal places
      System.out.println("Your weight is "+String.format("%.4f",weightOnMars)+" lbs on Mars");

        System.out.println("Converting the variable into integer");
        int weightOnMarsInt = (int)weightOnMars;
        System.out.println(weightOnMarsInt);

        System.out.println("Converting the variable into char");
        char weightOnMarsChar = (char)weightOnMars;
        System.out.println(weightOnMarsChar);

        System.out.println("Converting the variable into Int and doing an operation on it");
        int newIntVariable = weightOnMarsChar * 2;
        System.out.println(newIntVariable);
 }

  }
 }

 }
```

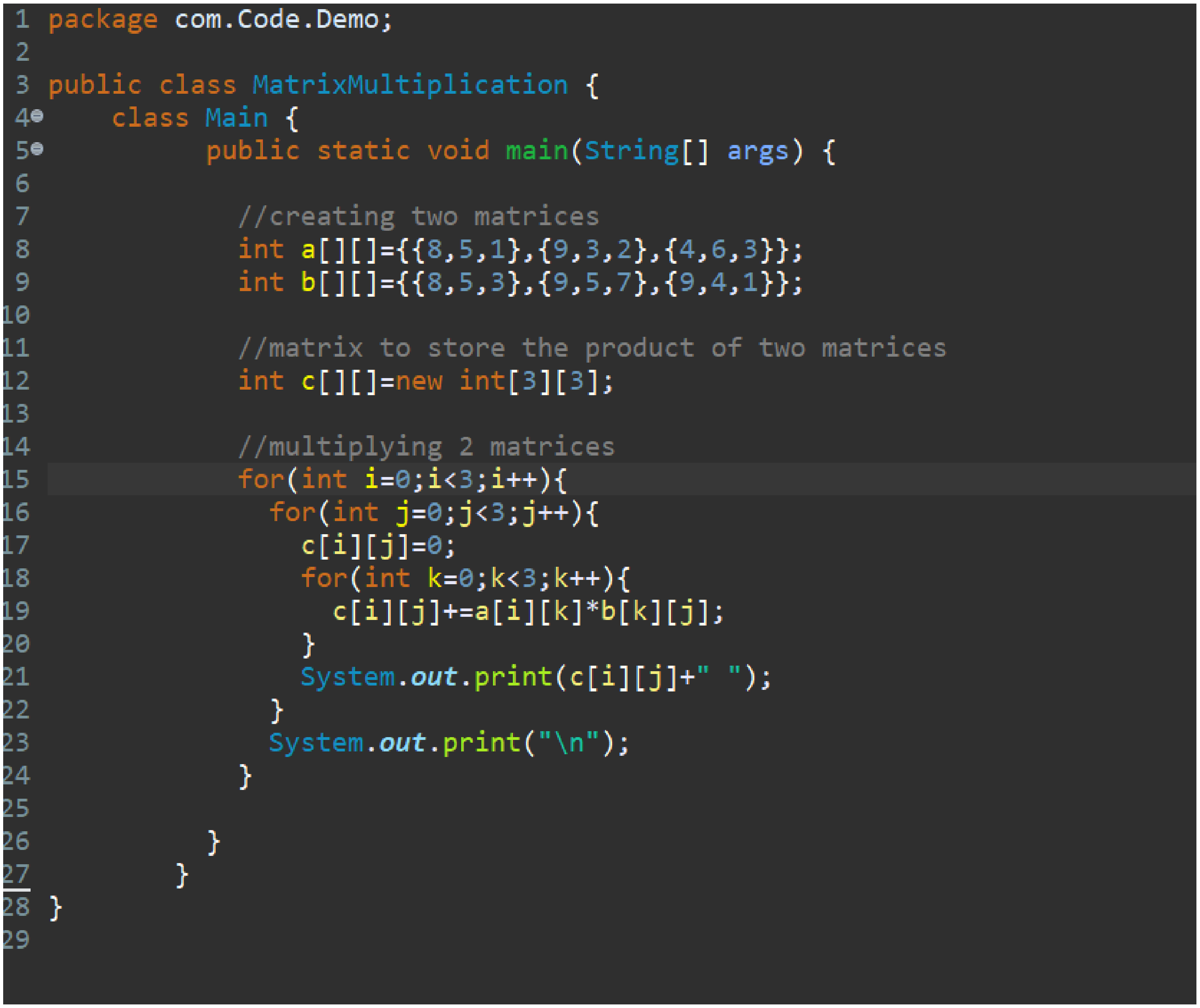
**Output :-**



```
How many pounds (lbs) do you weigh? 152
Your weight is 57.7600 lbs on Mars
Converting the variable into integer
57
Converting the variable into char
9
Converting the variable into Int and doing an operation
on it
114
```

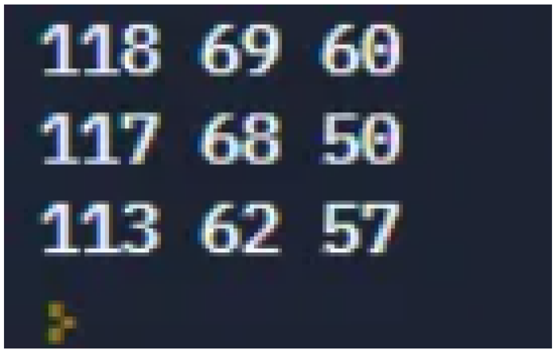
**5. Write a Java program of matrix multiplication?**

**PROGRAM :-**



```java
package com.Code.Demo;

public class MatrixMultiplication {
    class Main {
          public static void main(String[] args) {

            //creating two matrices
            int a[][]={{8,5,1},{9,3,2},{4,6,3}};
            int b[][]={{8,5,3},{9,5,7},{9,4,1}};

            //matrix to store the product of two matrices
            int c[][]=new int[3][3];

            //multiplying 2 matrices
            for(int i=0;i<3;i++){
              for(int j=0;j<3;j++){
                c[i][j]=0;
                for(int k=0;k<3;k++){
                  c[i][j]+=a[i][k]*b[k][j];
                }
                System.out.print(c[i][j]+" ");
              }
              System.out.print("\n");
            }

          }
        }
}
```

**OUTPUT :**



```
118 69 60
117 68 50
113 62 57
>
```

6 . The below mentioned points are CRS (Customer Requirement Specification)

▪ Customer secured entry

▪ Search the products needed

▪ Need to add that into my favorites

▪ Selected products need to deliver to the customer For the above-mentioned CRS, first you need to create SRS/FS and write the producers using V-model

Answer :-

**An SRS can be simply summarized into four Ds:**

1. **Define your product's purpose.**
2. **Describe what you're building.**
3. **Detail the requirements.**
4. **Deliver it for approval.**

We want to DEFINE the purpose of our product, DESCRIBE what we are building, DETAIL the individual requirements, and DELIVER it for approval. A good SRS document will define everything from how software will interact when embedded in hardware to the expectations when connected to other software. An even better SRS documents also account for real-life users and human interaction.

The best SRS documents define how the software will interact when embedded in hardware — or when connected to other software. Good SRS documents also account for real-life users.

**Why Use an SRS Document?**

This layout not only keeps your teams in sync but it also ensures that each requirement is hit. It can ultimately help you make vital decisions on your product's lifecycle, such as when to retire an obsolete feature.

The time it takes to write an SRS is given back in the development phase. It allows for better understanding or your product, team, and the time it will take to complete.

**Software Requirements Specification vs. System Requirements Specification**

A software requirements specification (SRS) includes in-depth descriptions of the software that will be developed.

A system requirements specification (SyRS) collects information on the requirements for a system.

“Software” and “system” are sometimes used interchangeably as SRS. But, a software requirement specification provides greater detail than a system requirements specification.

## How to Write an SRS Document

Writing an SRS document is important. But it isn’t always easy to do.

Here are five steps you can follow to write an effective SRS document.

### 1. Define the Purpose With an Outline (Or Use an SRS Template)

Your first step is to create an outline for your software requirements specification. This may be something you create yourself. Or you may use an existing SRS template.

If you’re creating this yourself, here’s what your outline might look like:

1. Introduction

    1.1 Purpose

    1.2 Intended Audience

    1.3 Intended Use

    1.4 Scope

    1.5 Definitions and Acronyms

2. Overall Description

2.1 User Needs

2.2 Assumptions and Dependencies

3. System Features and Requirements

3.1 Functional Requirements

3.2 External Interface Requirements

3.3 System Features

3.4 Nonfunctional Requirements

This is a basic outline and yours may contain more (or fewer) items. Now that you have an outline, lets fill in the blanks.

*Download a white paper on best practices for writing requirements >>*

## 2. Define your Product’s Purpose

This introduction is very important as it sets expectations that we will hit throughout the SRS.

Some items to keep in mind when defining this purpose include:

*Intended Audience and Intended Use*

Define who in your organization will have access to the SRS and how they should use it. This may include developers, testers, and project managers. It could also include stakeholders in other departments, including leadership teams, sales, and marketing. Defining this now will lead to less work in the future.

*Product Scope*

What are the benefits, objectives, and goals we intend to have for this product? This should relate to overall business goals, especially if teams outside of development will have access to the SRS.

*Definitions and Acronyms*
It's important to define the risks in the project. What could go wrong? How do me mitigate these risks? Who is in charge of these risk items?

For example, if the failure of a medical device would cause slight injury, that is one level of risk. Taking into account the occurrence level and the severity, we can come up with a strategy to mitigate this risk.

## 3. Describe What You Will Build

Your next step is to give a description of what you're going to build. Is it a new product? Is it an add-on to a product you've already created? Is this going to integrate with another product? Why is this needed? Who is it for?

Understanding these questions on the front end makes creating the product much easier for all involved.

*User Needs*
Describe who will use the product and how. Understanding the user of the product and their needs is a critical part of the process.

Who will be using the product? Are they a primary or secondary user? Do you need to know about the purchaser of the product as well as the end user? In medical devices, you will also need to know the needs of the patient.

*Assumptions and Dependencies*
What are we assuming will be true? Understating and laying out these assumptions ahead of time will help with headaches later. Are we assuming current technology? Are

we basing this on a Windows framework?  We need to take stock of these assumptions to better understand when our product would fail or not operate perfectly.

Finally, you should note if your project is dependent on any external factors. Are we reusing a bit of software from a previous project? This new project would then depend on that operating correctly and should be included.

## 4. Detail Your Specific Requirements

In order for your development team to meet the requirements properly, we MUST include as much detail as possible. This can feel overwhelming but becomes easier as you break down your requirements into categories. Some common categories are:

*Functional Requirements*

Functional requirements are essential to your product because, as they state, they provide some sort of functionality.

Asking yourself the question “does this add to my tool’s functionality?” Or “What function does this provide?” can help with this process. Within Medical devices especially, these functional requirements may have a subset of risks and requirements.

You may also have requirements that outline how your software will interact with other tools, which brings us to external interface requirements.

*External Interface Requirements*

External interface requirements are specific types of functional requirements. These are especially important when working with embedded systems. They outline how your product will interface with other components.

There are several types of interfaces you may have requirements for, including:

- User
- Hardware
- Software

- Communications

*System Features*
System features are types of functional requirements. These are features that are required in order for a system to function.

*Other Nonfunctional Requirements*
Nonfunctional requirements can be just as important as functional ones.

These include:

- Performance
- Safety
- Security
- Quality

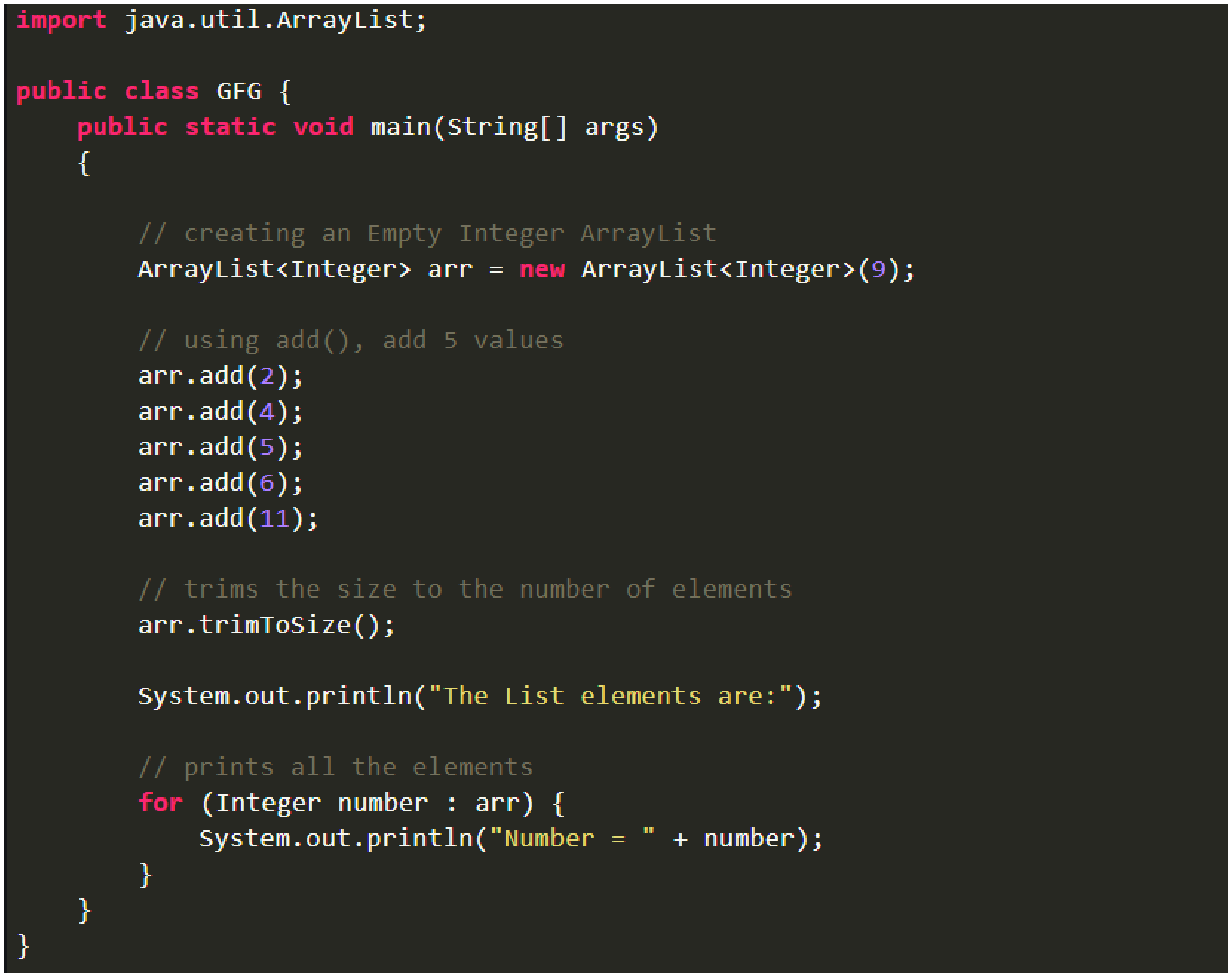
The importance of this type of requirement may vary depending on your industry. In the medical device industry, there are often regulations that require the tracking and accounting of safety.

*IEEE also provides guidance for writing software requirements specifications, if you're a member.*

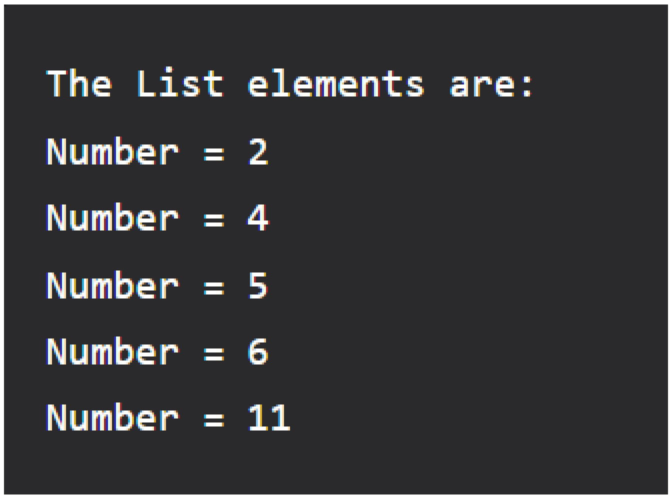
## 5. Deliver for Approval
We made it! After completing the SRS, you'll need to get it approved by key stakeholders. This will require everyone to review the latest version of the document.

7. Implement the Array list concepts for the questions given below. Write a Java program, A. to trim the capacity of an Array list the current list size. B. to increase the size of an Array list. C. to replace the second element of an Array List with the specified element. D. to print all the elements of an Array List using the position of the elements.

A. to trim the capacity of an Array list the current list size.

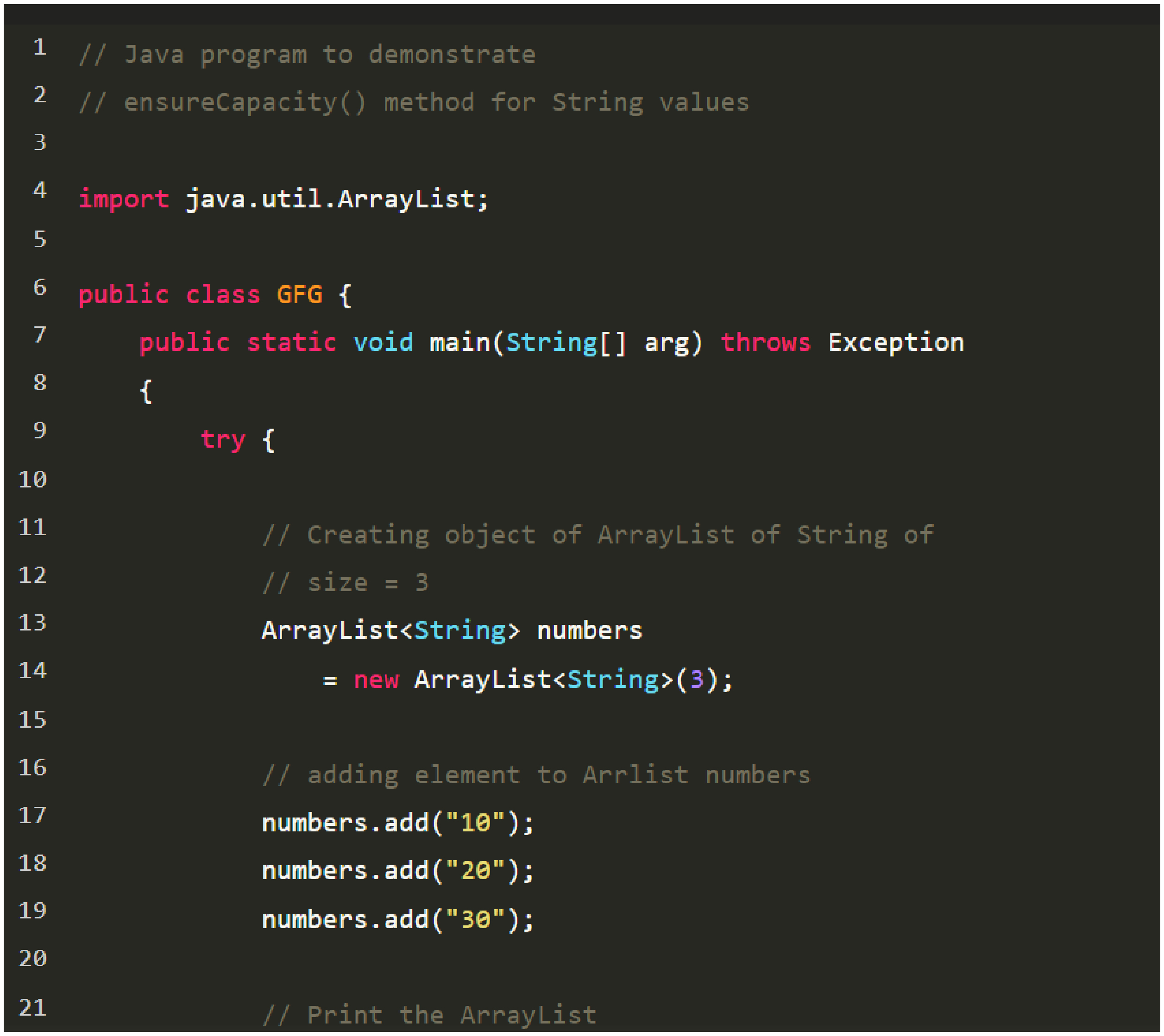


```java
import java.util.ArrayList;

public class GFG {
    public static void main(String[] args)
    {

        // creating an Empty Integer ArrayList
        ArrayList<Integer> arr = new ArrayList<Integer>(9);

        // using add(), add 5 values
        arr.add(2);
        arr.add(4);
        arr.add(5);
        arr.add(6);
        arr.add(11);

        // trims the size to the number of elements
        arr.trimToSize();

        System.out.println("The List elements are:");

        // prints all the elements
        for (Integer number : arr) {
            System.out.println("Number = " + number);
        }
    }
}
```

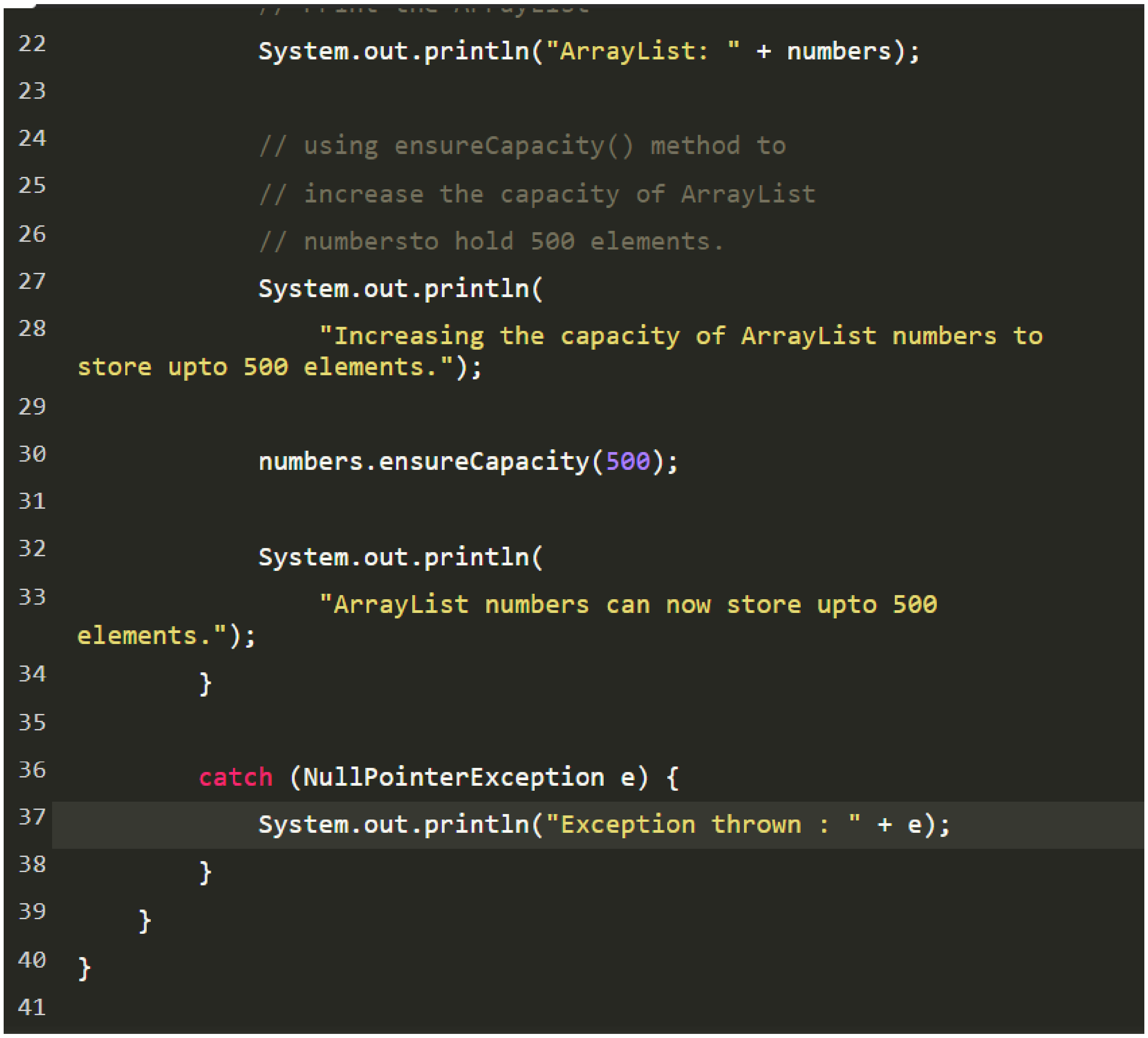


```
The List elements are:
Number = 2
Number = 4
Number = 5
Number = 6
Number = 11
```

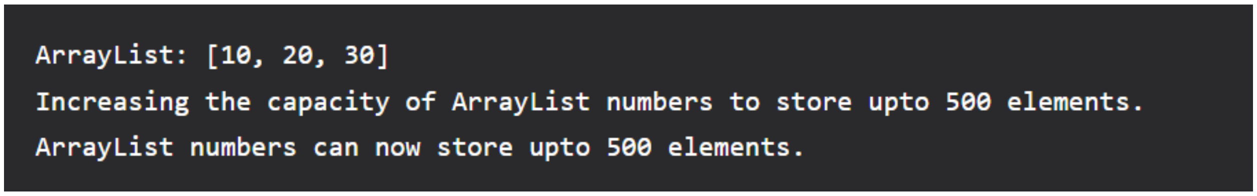
**b). to increase the size of an Array list**



```java
// Java program to demonstrate
// ensureCapacity() method for String values

import java.util.ArrayList;

public class GFG {
    public static void main(String[] arg) throws Exception
    {
        try {

            // Creating object of ArrayList of String of
            // size = 3
            ArrayList<String> numbers
                = new ArrayList<String>(3);

            // adding element to Arrlist numbers
            numbers.add("10");
            numbers.add("20");
            numbers.add("30");

            // Print the ArrayList
```

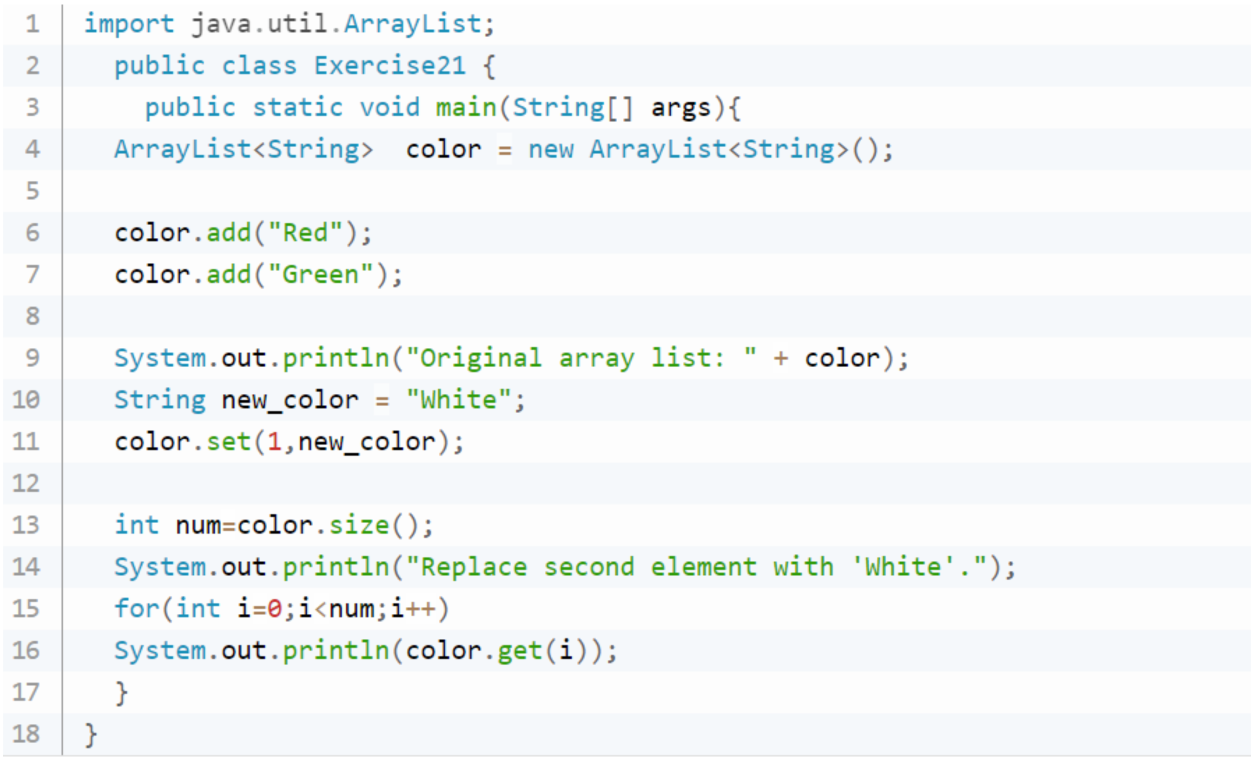


```java
            System.out.println("ArrayList: " + numbers);

            // using ensureCapacity() method to
            // increase the capacity of ArrayList
            // numbersto hold 500 elements.
            System.out.println(
                "Increasing the capacity of ArrayList numbers to store upto 500 elements.");

            numbers.ensureCapacity(500);

            System.out.println(
                "ArrayList numbers can now store upto 500 elements.");
        }

        catch (NullPointerException e) {
            System.out.println("Exception thrown : " + e);
        }
    }
}
```

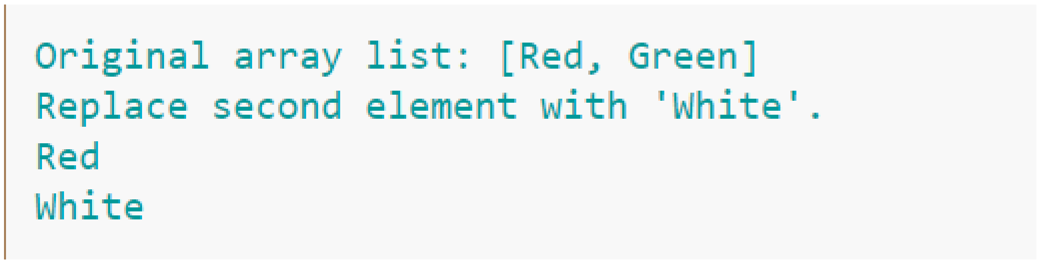


```
ArrayList: [10, 20, 30]
Increasing the capacity of ArrayList numbers to store upto 500 elements.
ArrayList numbers can now store upto 500 elements.
```

**C. to replace the second element of an Array List with the specified element.**



```java
import java.util.ArrayList;
  public class Exercise21 {
    public static void main(String[] args){
  ArrayList<String>  color = new ArrayList<String>();

  color.add("Red");
  color.add("Green");

  System.out.println("Original array list: " + color);
  String new_color = "White";
  color.set(1,new_color);

  int num=color.size();
  System.out.println("Replace second element with 'White'.");
  for(int i=0;i<num;i++)
  System.out.println(color.get(i));
  }
}
```



```
Original array list: [Red, Green]
Replace second element with 'White'.
Red
White
```

**D. to print all the elements of an Array List using the position of the elements.**

```java
import java.util.ArrayList;
  public class Exercise22 {
    public static void main(String[] args) {
  ArrayList <String> c1 = new ArrayList <String> ();
  c1.add("Red");
  c1.add("Green");
  c1.add("Black");
  c1.add("White");
  c1.add("Pink");
  System.out.println("\nOriginal array list: " + c1);
  System.out.println("\nPrint using index of an element: ");
  int no_of_elements = c1.size();
  for (int index = 0; index < no_of_elements; index++)
   System.out.println(c1.get(index));
 }
}
```

```
Original array list: [Red, Green, Black, White, Pink]

Print using index of an element:
Red
Green
Black
White
Pink
```

**8. Which one is better to implement in Java? extending Thread class or implementing Runnable?**

Answer:-The Java Thread itself implements a Java Runnable! and according to most of the experts over Internet, implements Runnable is preferred over extends Thread! even though we cannot use utilize Runnable in the sense of thread with out the Thread class!

Then why do we prefer to implement Runnable over extending Thread since in both cases the actual thread is stated by calling a Thread implemented method (i.e. start() or run()) although in case of Thread we are not truly "extending" the functionality of Thread by merely overriding the run() method?

**9.)Write a java program, the user has created an account , when the user logged in to that account, if the user enter the correct data it should give the message as “Successfully logged in “ or Need to provide the correct data to ring that msg. (Need to get input at runtime).**

```
package assesment_exam;



import java.util.Scanner;



public class User_Acc {



	public static void main(String[] args) {
```

```
Scanner sc = new Scanner(System.in);

int id = 1001;

String password = "idfc123";

// Taking input

System.out.println("Enter user id: ");

int a = sc.nextInt();

System.out.println("Enter password:- ");

String name = sc.next();

if(a==id && name.equals(password)) {

        System.out.println("Successfully logged in");

}else {

        System.out.println("Please enter correct data !!");
        }

        sc.close();

    }

}
```