OOP GD

GD Script
Introduction
Atharva R: Good Afternoon everyone, This is group 4 from batch 3 presenting you the topic for our group discussion “Difference Between 1) Throw and Throws 2) Final, Finally and Finalize”

Atharva S.: This keywords are related to exception so we will discuss about exception first then we will discuss about the keywords.
Exception: Exception is an event that disrupts the normal flow of execution of Program.
There are various reasons that causes the exception to occur while some of them can be Invalid user input, Opening unavailable file etc.

Harshal : Basically there are two types of exceptions are 1)Built in exception 2)User Defined exception

1) Built in Exception
As the name suggests these are the exceptions that are already available in Java libraries. These exceptions are able to define the error situation so that we can understand the reason of getting this error. It can be categorized into two broad categories, i.e., checked exceptions and unchecked exception.

Pratham: i.checked exception: Checked exception are those that are checked during the compile time by the compiler. The compiler ensures whether the programmer handles the exception or not. The programmer should have to handle the exception; otherwise, the system has shown a compilation error.
Can anyone please mentions eg. for checked exceptions
Checked Exception example:
Bhavin: SQLException info
Uma: IOException
I would like to add some examples of checked exceptions , common examples of checked exceptions are SQLException, IOException etc
where as IOException occurs When user tries to give wrong input
For e.g if pro asks for say roll_no and its datatype is int but incase user tries to give data other than in int format it generates an IOexception

Rucha: ii. unchecked exception: Unchecked exception are those that are not checked during compile time by the compiler rather they occur in run time.
Can anyone add the example for this types of exceptions?

Some examples of unchecked exceptions can be
Shraddha: ArithmeticException, info Basically if we talk about ArithmeticException Divide by zero is common Exception which occurs when users tries to divide the any no with zero.

Mansi: NullPointerException, when user tries to access any array index out of bounds ArrayIndexOutOfBoundsException occurs.
Ig we have much more seen about Builtin Exceptions let's discuss now about User Defined Exception

Shashank: ArrayIndexOutOfBoundsException etc info when user tries to access any array index out of bounds ArrayIndexOutOfBoundsException occurs.
Ig we have much more seen about Builtin Exceptions let's discuss now about User Defined Exception

Atharva R. : 2) User Defined Exception:
Built-in exceptions are restricted to trigger on some predefined conditions. But in Java, we can write our own custom exception i.e user can create exceptions as per programs requirement.
Chaitanya : It can be done by creating a class which extends the Exception class which is parent or base class of all other Exception classes..

Shraddha: So the exception causes the disruption of normal flow of execution so to handle the exception so that the normal flow of the application can be maintained we use exception handling.

Harshal : There are various ways through which the exception handling can be achieved such as try catch block, throw, throws, etc. So now let us discuss about Throw and Throws

- **Pratham: Throw:** The throw keyword is used to manually throw the exception.We can also say that the throw keyword is used to throw an exception explicitly. Here we specify the exception object which is to be thrown. Exception has some message with it that provides the error description.
- **Mansi:** We can either throw a checked or unchecked exception using throw keyword but most of the time it is used to throw the custom exception.
  We can also define our own set of conditions and throw an exception explicitly using throw keyword. For example, we can throw ArithmeticException if we divide a number by another number. Here, we just need to set the condition and throw exception using throw keyword.

  Can anyone add syntax for it?

  Shashank : Syntax: throw new exception_class(“error message”);

  Example: throw new IOException(“This is a device error”);
- **Atharva S: Throws:**Throws keyword is used to declare the exception. It is usually declared with the signature of method to indicate that this method might throw one of the listed type exceptions. The caller to these methods has to handle the exception using a try-catch block.

Rucha: In a program, if there is a chance of raising an exception then the compiler always warns us about it and compulsorily we should handle that checked exception. Otherwise we will get a compile time error saying unreported exception XXX must be caught or declared to be thrown. To prevent this compile time error we use throws keyword.

Uma: We can use throws keyword to delegate the responsibility of exception handling to the caller (It may be a method or JVM) then the caller method is responsible to handle that exception.
Can anyone add syntax for it?

Chaitanya: Syntax: void method_1() throws Exception;

Bhavin: For Example, if a method in our program want to access a text file and read its contents then we will use FileReader class in that method and When we run the program, if the mentioned text file does not exist, FileReader throws a FileNotFoundException which extends the IOException class. And output will be FileNotFoundException; must be caught or declared to be thrown
So we must specify throws IOException clause with method signature so that methods further up in the call stack can handle them.

Atharva R.: Another example is when we are doing division in any method then there may be the chances that the divisor may be 0 and it can throw ArithmeticException so we must add throws ArithmeticException at the end of the method signature.
Example: void division_method(int a, int b) throws ArithmeticException;

Shraddha : Sometimes people get confused between throw and throws so let us discuss about the difference between throw and throws

SR.No	Basics of difference	throw	throws
1	**Pratham**: definition	**Pratham:** Java throw keyword is used to throw an exception explicitly in the code, inside the function or the block of code.	**Harshal**: Java throws keyword is used in the method signature to declare an exception which might be thrown by the function while the execution of the code

2	**Atharva S**. Internal Implementations	**Atharva S** it is allowed to throw an only single exception at a time i.e we cannot throw multiple exceptions with throw keyword.	**Shashank** On another hand we can declare multiple exceptions with throws keyword that could get thrown by the function where throws keyword is used.
3	**Rucha** Type of Exception	**Rucha** With throw keyword we can propagate only unchecked exception i.e checked exception cannot be propagated using throw.	**Mansi** throws keyword both checked and unchecked exceptions can be declared and for the propagation checked exception must use throws keyword followed by specific exception class name.
4	**Shraddha** Declaration	**Shraddha:** throw is used within the method.	**Chaitanya:** throws is used with the method signature.

Chaitanya: There are other keywords as well where people also gets confused some of them are Final, Finally and Finalize

- **Atharva R.** Final: Final is similar to const in c or c++. It means once a variable is declared having final as a prefix then its value cannot be changed in future. If a variable is declared as final with no initialization then that value can only be initialized in the constructor.
  Final keyword can be used with
  1) Variable
  2) Method
  3) Class

I. **Rucha** Variable: When we use the Final keyword with the variable then the value of that variable cannot be modified further. It will remain constant throughout the program
For example If we want to set a fix value to speedlimit can should not be changed then we can declare speedlimit variable as final because final variable once assigned a value can never be changed.

II. **Uma** Method: When we use the Final keyword with the method then that method cannot be overridden.

III. **Pratham** Class:When we use Final keyword with the class then any class cannot be inherited from that class
Advantage:
Bhavin:
Final keyword improves performance. ...
Final variables are safe to share in a multi-threading environment without additional synchronization overhead.
Final keyword allows JVM to an optimized method, variable or class.

- **Mansi** Finally: Finally keyword is use in Exception concept. It is used along with try-catch block. If we need to execute a code even when the exception is generated or not then that code can be put in finally block.

- **Shashank** Finalize: It is a method of Object super class which is protected by default. We need to to override the Finalize() method. This method is used to release the resources that is allocated to an unused object before the unused object is removed by garbage collector in java. SO to get clear info lets compare these three with different parameter

Keyword	Final	Finally	Finalize
Defination	**Atharva S.** final is the keyword and access modifier which is used to apply restrictions on a class, method or variable.	**Harshal** finally is the block in Java Exception Handling to execute the important code whether the exception occurs or not.	**Pratham** finalize is the method in Java which is used to perform clean up processing just before object is garbage collected.
Applicable to	**Atharva R.** Final keyword is used with the classes, methods and variables.	**UMA** Finally block is always related to the try and catch block in exception handling.	**Chaitanya**: finalize() method is used with the objects.
Functionality	**Mansi** (1) Once declared, final variable becomes constant and cannot be modified. (2) final method cannot be overridden by sub class. (3) final class cannot be inherited.	**Shraddha** (1) finally block runs the important code even if exception occurs or not. (2) finally block cleans up all the resources used in try block	**Rucha** finalize method performs the cleaning activities with respect to the object before its destruction.
Execution	**Shashank** Final method is executed only when we call it.	**Bhavin** Finally block is executed as soon as the try-catch block is executed. Its execution is not dependent on the exception. Regardless of occurrence of exception finally block is always	**Atharva R**. finalize method is executed just before the object is destroyed.

		executed	

Harshal So we have discussed all about throw, throws keyword as well as final, finally and finalize keywords and we can conclude from the discussion that though the pronunciation of the keywords may be similar but the meaning and usage of the keywords differ a lot.