

Objective:

- 1.B(Arithmetic Exception)
- 2.B(start exception finally end)
- 3.A(car maruthitata)
- 4.A(myException)
- 5.A
- 6.A(compilation error)
- 7.D(range)
- 8.A
- 9.B(init.service,destroy)
- 10.B(hover)

Subjective::

🚩 1.Difference between array list and linked list.

A)Arraylist: 1.In first place array is a fixed size. It means, we will declare the number of elements we use at the time of declaring the array. Once we declare the array we cannot change. So, if we want to insert or delete an element from the array, we use the concept called Dynamic array. By using the concept of dynamic array we can change the size of the array by insertion ,deletion etc.

for remove an element from specific index::

```
if (count > 0) {  
    for (int i = index; i < count - 1; i++)  
    {  
        array[i] = array[i + 1];  
    }  
    array[count - 1] = 0;  
    count--;  
}
```

```
}
```

for inserting an element::

```
Inta[] = {1,2,3};
```

```
System.out.println("Array:"+Arrays.toString(a)); o/p:[1,2,3]
```

```
ArrayList<Integer>arrayList = new ArrayList<Integer>();
```

```
arrayList.add(4);
```

```
arrayList.add(5); o/p:[1,2,3,4,5]
```

```
a = arrayList.toArray(a);
```

```
System.out.println("Array after inserting element: "+Arrays.toString(a));
```

2.Arraylist consists of duplicate values. That means if we give one value more than one time ,it will give the same number of times.

```
Eg:Listlst = new ArrayList<>(2);
```

```
lst.add(0);
```

```
lst.add(1); o/p:[0,1,2,1,4,5]
```

```
lst.add(2);
```

```
lst.add(1);
```

```
lst.add(4);
```

```
lst.add(5);
```

```
System.out.println(lst);
```

3.We also can access values using get method by passing index as a parameter.

```
Eg:Listlst = new ArrayList<>(2);
```

```
lst.add(0);
```

```
lst.add(1); o/p: 1
```

```
lst.add(2);
```

```
lst.add(1);
```

```
lst.add(4);
```

```
lst.add(5);
```

```
System.out.println(lst.get(3));
```

LinkedList: 1. Java LinkedList class can contain duplicate elements.

```
Ex::int main()
```

```
{
```

```
    Node* head = NULL;
```

```
insert(&head, 5);
```

```
insert(&head, 7);          o/p:2
```

```
insert(&head, 5);
```

```
insert(&head, 1);
```

```
insert(&head, 7);
```

```
cout<<countNode(head);
```

```
return 0;
```

```
}
```


2. linked list means one element is connected to another element. So, the linked list class can behave as both list and queue. As it implements the interfaces of deque and lists.

```
Ex:: Queue<Integer> q= new LinkedList<>();
```

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

```
q.add(i);
```

```
System.out.println("Elements of queue "+ q);
```

 2. Error page means, When we see a "404 error" or "page not found" on a page, where we wrote a program for the appearance of webpage on the back. The page says that the user reached the domain requested, but the URL provided with no information. If we want to create an error page, we have to set page directive attribute isErrorPage value to true.

```
<html>
```

```
<body>


<%@ page isErrorPage="true" %>


error:

<%= exception %>


</body>

</html>
```

 3. Different types of tags:: The types of tags present in jsp are
1. Expression tags 2. Scriptlet tags 3. Declaration tags 4. Directive tags

 5. The Fail Fast iterators immediately throw `ConcurrentModificationException` in case of structural modification of the collection. Structural modification means adding, removing, updating the value of an element in a data collection while another thread is iterating over that collection.


OverCome:- The Fail Safe iterators are just opposite to Fail Fast iterators; unlike them, A fail-safe iterator does not throw any exceptions unless it can handle if the collection is modified during the iteration process. This can be done because they operate on the copy of the collection object instead of the original object.

 6. Difference between
`getSession()` : Returns the current session associated with this request, or if the request does not have a session, creates one.

`getSession(true)` : Returns the current `HttpSession` associated with this request, if there is no current session, returns a new session

`getSession(false)` : Returns the current `HttpSession` associated with this request, if there is no current session, returns null.

`getSession()` is preferable because it checks and creates the new one.

 7) Memory Leakage::
A Java memory leak happens when an application unintentionally (due to logical errors in code) holds on to object references that are no longer required. These unintentional object references prevent the built-in Java garbage collection mechanism from freeing up the memory consumed by these objects.

Prevention:

1. When have to release the session when it is no longer needed. Use the `HttpSession.invalidate()` for this.

2.Keep the time-out time low for each session.

3.Store only the necessary data in your HttpSession.

4.Avoid using string concatenation. Use StringBuffer'sappend() method because the string is an unchangeable object while string concatenation creates many unnecessary objects. A large number of temporary objects will slow down performance.

5.As much as possible, you should not create HttpSession on your jsp page. You can do this by using the page directive <%@page session="false"%>.

8)Yes, we can declare a class as final.Final is a non-access modifier.When we declare a class as a final it is known as final class.Once we declared as final we cannot change anythingThe final class can't be inherited.There are two uses of the final class
1) to prevent inheritance

Eg:

```
final class Car{
```

```
    Int tyres;
```

```
    Int gears;
```

```
}
```

```
Class brezza extends Car{
```

```
    Int brake;
```

```
}
```

Output:compile error

2)The second use of final with classes is to create an immutable class like the predefined String class.you can't make a class immutable without making it final.

10)Clone in Java refers to the creation of an Exact copy of an object. It creates a new instance of the class of the Existing object and initializes all its attributes with exact Values of the corresponding attributes of this object. there is no operator to create a copy of an object.we have a method to create the clone of the object called Clone() method. The java.lang.Cloneable interface must be implemented by the class whose object is going to create .if we don't implement the cloneable interface it generates an exception "CloneNotsupportedException".

If We use Clone() method it saves the extra processing task for creating the exact copy of an object.if we use new keyword for creating object it takes a lot of processing time that's why we use object cloning method.

Eg;

```
class Student implements Cloneable{

int id;

String Name;

Student(intid,String Name){

this.id=id;

this.Name=Name;

}

public Object clone()throws CloneNotSupportedException{

returnsuper.clone();

}

public static void main(String args[]){

try{

Student t1=new Student(1,"Vaishnavi");

Student t2=(Student)t1.clone();

System.out.println(t1.rollno+" "+t1.name);

System.out.println(t2.rollno+" "+t2.name);

}catch(CloneNotSupportedException c){}

}

}
```

Programs::

1)package com.java;

importjava.util.Scanner;

```
public class Sum_of_two_no {
```

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

```

Scanner sc=new Scanner(System.in);

System.out.print("Enter the Size of Array : ");

intnum = sc.nextInt();

intarr[] = new int[num];

for(int i=0; i<num; i++)

arr[i] = sc.nextInt();

System.out.println("Enter the number ");

int n = sc.nextInt();

int flag=0;

for(int i=0; i<num; i++)

{

    for(int j = 0; j<num; j++)

    {

        if(arr[i]+arr[j] == n) {

            flag++;

            System.out.println("{}+arr[i]+", "+arr[j]+");

        }

    }

}

if(flag==0)

    System.out.println("can't get value for given sum ");

}

}

```

2)package demo;

```

import java.util.*;

import java.io.*;

class test{

static int MAX = 256;

private boolean checkString(String str1, String str2)

    {

int[] count = new int[MAX];

char []str3 = str1.toCharArray();

for (int i = 0; i < str3.length; i++)

count[str3[i]]++;

char []str4 = str2.toCharArray();

for (int i = 0; i < str4.length; i++) {

if (count[str4[i]] == 0)

return false;

count[str4[i]]--;

    }

return true;

    }

public static void main(String args[])

    {

        Scanner sc= new Scanner(System.in);

System.out.println("Enter First String : ");

        String str1 = sc.nextLine();

System.out.println("Enter Second String : ");

        String str2 = sc.nextLine();

testobj = new test();

```



```
boolean flag = obj.checkString(str1,str2);
```

```
System.out.println(flag);
```

```
}
```

```
}
```