Dt : 31/7/2023

Summary of Control Statements:

1.Selection Statements:

=>Types:

(a)simple if

=>The process of declaring if-block without else-block is known as "simple if"

(b)if-else

=>we use if-else block in the program when we want to check true or false

(c)Nested if(Inner if)

=>The process of declaring "if" inside the "if" is known as Nested if or "Inner if"

(d)Ladder if-else

=>The process of inter-linking more number of if-else blocks is known as "Ladder if-else"

(e)switch-case

=>we use switch-case in the program when we want to select one from multiple cases or options.

2.Iterative Statements:

=>Types:

(a)while loop

=>In while looping structure the condition is checked first and if the condition is true then loop-body is executed,this process is repeated until the condition is false.

(b)do-while loop

=>In do-while looping structure the loop-body is executed first and then condition is checked,this process is repeated until the condition is false.

(c)for loop

=>for-loop is more simple in representation when compared to while and do-while loops,because Initialization,Condition and Incre/Decre are declared in the same line separated by SemiColon.

3.Branching Statements:

=>Types:

(a)break

=>break is used to stop the switch-case execution.

=>break is also used to stop iterative statements.

(b)continue

=>'continue" will skip the below lines from the iteration.

(c)return

=>'return' statement is used to return the value after method execution and the returned value will come back to the method call.

(d)exit

=>"exit" will stop the program execution or will terminate the program execution

==========

faq:

wt is the diff b/w

(i)while loop

(ii)do-while loop

=>In while-loop condition is checked first then loop-body is executed,but in do-while loop the loop-body is executed first and then the condition is checked.

=>In while-loop the body is not executed for false-condition,but in do-while loop the body is executed once for the false condition.

Note:

=>do-while loop in applications will waste the execution time in executing body for false condition and degrade the performance of an application.

==========

Note:

=>In realtime for-loop is used for Strings and Arrays,because they use index.

==========

Assignment:(Solution)

Update above program by displaying result as "Fail",if any one Subject marks are

in b/w 0 to 34

Program : StuMainClass.java(Modifed program)

```
import java.util.Scanner;
class StudentResult
{
	String res(float pr,boolean p)
	{
		if(p)
		{
			return "Fail";
		}
		else if(pr>=70 && pr<=100)
		{
			return "Distinction";
		}
		else if(pr>=60 && pr<70)
		{
			return "FirstClass";
		}
```

```
            else if(pr>=50 && pr<60)
            {
                return "SecondClass";
            }
            else if(pr>=35 && pr<50)
            {
                return "ThirdClass";
            }
            else
            {
                return "Fail";
            }
        }
}
class Percentage
{
        float per(int totMarks)
        {
            float p = (float)totMarks/6;
            return p;
        }
}
```

```
class StuMainClass
{
        public static void main(String[] args)
        {
                Scanner s = new Scanner(System.in);
                int i=1,totMarks=0;
                boolean p=false;
                while(i<=6)
      {
                  System.out.println("Enter marks of sub-"+i);
                  int sub = s.nextInt();
                  if(sub<0 || sub>100)
                        {
                          System.out.println("Invalid marks...");
                          continue;
                        }
                 if(sub>=0 && sub<=34)
                        {
                          p=true;
                        }
                  totMarks=totMarks+sub;
        i++;
```

```
    }
            System.out.println("TotalMarks:"+totMarks);
            Percentage ob = new Percentage();
            float pr = ob.per(totMarks);
    System.out.println("Percentage:"+pr);
            StudentResult sr = new StudentResult();
            String result = sr.res(pr,p);
            System.out.println("Result:"+result);
      }
}
```

==========

Note:

=>when we read String-data after reading numeric-data,the String-data reading

will be Skipped from Console-Input.

=>This can be overcomed using parse-methods,and these parse-methods will convert

String-data into numeric data.

=>The following are some important methods to read numric data:

```
byte var = Byte.parseByte(s.nextLine());
short var = Short.parseShort(s.nextLine());
int var = Integer.parseInt(s.nextLine());
```

```
   long var = Long.parseLong(s.nextLine());
   float var = Float.parseFloat(s.nextLine());
   double var = Double.parseDouble(s.nextLine());
```

Ex : DemoMethods7.java

```
import java.util.Scanner;
class Customer
{
        //Instance Variables
        int custNo;
        String custName;
   long phNo;
        void getCustomer()
        {
                System.out.println("****CustomerDetails****");
                System.out.println("CustNumber:"+custNo);
                System.out.println("CustomerName:"+custName);
                System.out.println("CustomerPhNo:"+phNo);
        }
}
class DemoMethods7
{
```

```
	public static void main(String[] args)
	{
		Scanner s = new Scanner(System.in);
		Customer c = new Customer();
		System.out.println("Enter the CustNo:");
		c.custNo = Integer.parseInt(s.nextLine());
		System.out.println("Enter the CustName:");
		c.custName = s.nextLine();
		System.out.println("Enter the CustPhNo:");
		c.phNo = s.nextLong();
		c.getCustomer();
	}
}
```

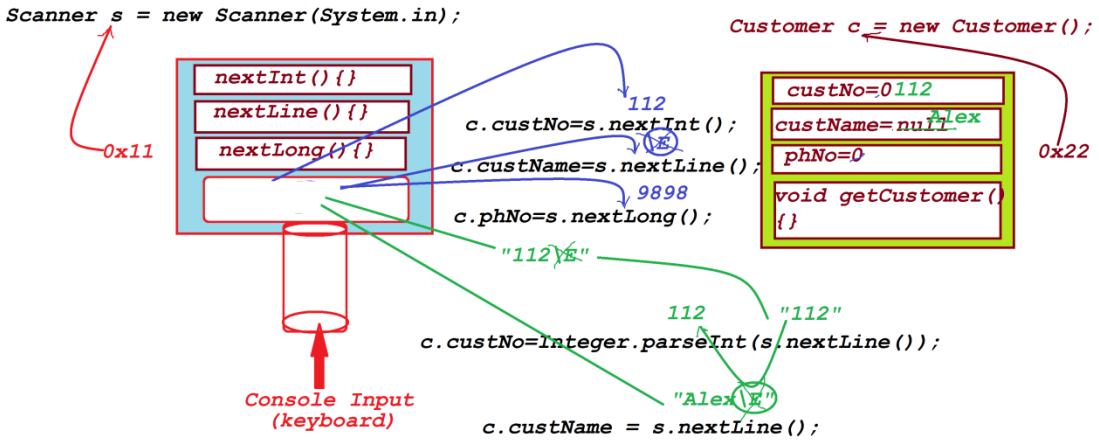
o/p:

```
Enter the CustNo:
112
Enter the CustName:
Alex
Enter the CustPhNo:
9898981234
****CustomerDetails****
CustNumber:112
```

CustomerName:Alex

CustomerPhNo:9898981234

Diagram:



==========