********** OUTPUT **********

TEST SCENARIOS

```
palindromicDates("01/01/2020",100) → {"02/02/2020"}
```

The word palindrome appears before the full DDMMYYYY

Welcome to Online IDE!! Happy Coding :)
2020
Palindrome
02/02/2020
2021
Palindrome
12/02/2021
2022
Palindrome
22/02/2022
2023
2024
2025
2026
2027
2028
2029
2030
Palindrome
03/02/2030
2031
Palindrome
13/02/2031
2032
Palindrome
23/02/2032
2033
2034
2035
2036
2037
2038
2039
2040
Palindrome
04/02/2040
2041
Palindrome
14/02/2041
2042
Palindrome
24/02/2042

2043
2044
2045
2046
2047
2048
2049
2050
Palindrome
05/02/2050
2051
Palindrome
15/02/2051
2052
Palindrome
25/02/2052
2053
2054
2055
2056
2057
2058
2059
2060
Palindrome
06/02/2060
2061
Palindrome
16/02/2061
2062
Palindrome
26/02/2062
2063
2064
2065
2066
2067
2068
2069
2070
Palindrome
07/02/2070
2071
Palindrome
17/02/2071
2072
Palindrome
27/02/2072
2073
2074
2075
2076
2077
2078
2079
2080
Palindrome

08/02/2080
2081
Palindrome
18/02/2081
2082
Palindrome
28/02/2082
2083
2084
2085
2086
2087
2088
2089
2090
Palindrome
09/02/2090
2091
Palindrome
19/02/2091
2092
Palindrome
29/02/2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
Palindrome
10/12/2101
2102
Palindrome
20/12/2102
2103
Palindrome
30/12/2103
2104
2105
2106
2107
2108
2109
2110
Palindrome
01/12/2110
2111
Palindrome
11/12/2111
2112
Palindrome
21/12/2112
2113
Palindrome

31/12/2113
2114
2115
2116
2117
2118
2119
2120
Palindrome
02/12/2120

** Process exited - Return Code: 0 **

```
palindromicDates("01/01/1900", 124);
```

Welcome to Online IDE!! Happy Coding :)
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930

1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987

1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
Palindrome
10/02/2001
2002
Palindrome
20/02/2002
2003
2004
2005
2006
2007
2008
2009
2010
Palindrome
01/02/2010
2011
Palindrome
11/02/2011
2012
Palindrome
21/02/2012
2013
2014
2015
2016
2017
2018
2019
2020
Palindrome
02/02/2020
2021
Palindrome
12/02/2021
2022
Palindrome
22/02/2022
2023
2024

** Process exited - Return Code: 0 **

/
*** CODE **

```
/*
/*
Online Java - IDE, Code Editor, Compiler

Online Java is a quick and easy tool that helps you to build, compile, test your programs online.
*/


public class Main
{
	static int year;		// this is the year
	static String yearString;
	static String finalDate;   //this will be in format DD/MM/YYYY
	static String finalDatePalindrome;  // this will be DDMMYYYY to ensure palindrome check can be complete
	static Object days;  //object since it is extracted from enum value
	static Object month;  //object since it is extracted from enum value
	static Object objectLeapYearFebruaryDays;   //extracted from enum value
	static int intLeapYearFebruaryDays;  //stores 29 on leap year
	static boolean leapYear;  //check if leap year
	static String temp;
	static Object prefixZero = "0";   //added to form DD for days less than 10
	static int daysMonth;  //uses days Object from above
	static int monthcount; // ensures main do while loop is constrained
	static String formattedMonth;  //obtain correct format such 01, 02...

	static int ddInt;  // used to initiate starting date in for loop...
	static int mmInt;  // used to select month and set month

	static String ddString;  // linked to ddInt
	static String mmString; //linked to mmInt

	static boolean processedFirstMonth=false;  // used to control ddInt... Once first month is complete,
	//start date ddInt each month will be 1


   //method call to check if palindrome
   public static void checkPalindrome(String finalDate, String finalDatePalindrome)
   {

	//System.out.println("CHECKING!");
	String reverseString;

	int j=0;
```

```
    StringBuffer sb = new StringBuffer();

    for (int i=finalDatePalindrome.length()-1; i>=0; i--)
    {

        sb.append(finalDatePalindrome.charAt(i));
        j++;
    }


    reverseString=sb.toString();

    if(finalDatePalindrome.equalsIgnoreCase(reverseString))
    {
        System.out.println("Palindrome");
        System.out.println(finalDate);   // Date will be outputted DD/MM/YYYY

    }

}

// enum containing the days,  MM

enum Months
{
    JAN (31,1),
    FEB(28,2),
    MAR(31,3),
    APR(30,4),
    MAY(31,5),
    JUN(30,6),
    JUL(31,7),
    AUG(31,8),
    SEPT(30,9),
    OCT(31,10),
    NOV(30,11),
    DEC(31,12);

    public int days;
    public int month;

    Months (int days, int month)
    {
        this.days=days;
        this.month=month;
    }
}

public static void main(String[] args) {
    System.out.println("Welcome to Online IDE!! Happy Coding :)");

    palindromicDates("01/01/2020", 100);
}
```

```
   public static void palindromicDates (String date, int years)
   {
      int startPos = date.length()-4;  // this is marker to start the year  YYYY

      createDate(startPos, date, years);

   }


   public static void createDate(int startPos, String date, int years)
   {
      yearString = date.substring(startPos, date.length());

      year = Integer.valueOf(yearString);

      ddString = date.substring(0,2);
      ddInt = Integer.valueOf(ddString);

      mmString = date.substring(3,5);
      mmInt = Integer.valueOf(mmString)-1;    // this is reduced by 1 since it will be used in
contexts
                                  // involving zero index notation.

      boolean newYearRollOver=false;   // default value


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

      {
         System.out.println(year);  // to ensure memory limit not exceeded, this is only
statement printed to Screen
         //with exception of if palindrome successful
         newYearRollOver=false;
         leapYear=false;


         monthcount=0;

         do
         {

         // processing the months in the enum
         for (Months m: Months.values())
      {

         switch (mmInt)   // this is used to get the correct MM format.
                     // In hindsight, perhaps both parameters in JAN (31,1) should have been
string
         {
            case 0:
               formattedMonth = "01";
               //System.out.println(m.values()[mmInt].name());
```

```
    break;

case 1:
    formattedMonth = "02";
    //System.out.println(m.values()[mmInt].name());
    break;

case 2:
    formattedMonth = "03";
    //System.out.println(m.values()[mmInt].name());
    break;

case 3:
    formattedMonth = "04";
    //System.out.println(m.values()[mmInt].name());
    break;

case 4:
    formattedMonth = "05";
    //System.out.println(m.values()[mmInt].name());
    break;

 case 5:
    formattedMonth = "06";
    //System.out.println(m.values()[mmInt].name());
    break;

    case 6:
    formattedMonth = "07";
    //System.out.println(m.values()[mmInt].name());
    break;

    case 7:
    formattedMonth = "08";
    //System.out.println(m.values()[mmInt].name());
    break;

    case 8:
    formattedMonth = "09";
    //System.out.println(m.values()[mmInt].name());
    break;

    case 9:
    formattedMonth = "10";
    //System.out.println(m.values()[mmInt].name());
    break;

    case 10:
    formattedMonth = "11";
    //System.out.println(m.values()[mmInt].name());
    break;

    case 11:
```

```
        formattedMonth = "12";
        //System.out.println(m.values()[mmInt].name());
        break;

}

if (year%4==0)    //leap year
{
   objectLeapYearFebruaryDays = m.FEB.days;  // obtain number days from enum

   intLeapYearFebruaryDays = (int) (objectLeapYearFebruaryDays)+1;  //increase by 1

   leapYear=true;   //flag set to true

}

//System.out.println("This is the month:" + m.name());

days = m.days;


if (m.name()=="FEB" && leapYear)
{

   days = (Object) intLeapYearFebruaryDays;  // this is officially setting days as 29

}


daysMonth = (int) days;


if (processedFirstMonth)
{
   ddInt=1;   //will ensure second month will start from first day DD
}


//moving through days
for (int k=ddInt; k<=(int) daysMonth; k++)
{

   month = m.month;   //setting month object to  enum variable month.... i.e JAN, FEB

   temp="";   // clearing value from temp

   temp = Integer.toString(k);


   if (k<10)
   {
      //temp will store DD if  day of month is less than 10
      temp = prefixZero.toString() + Integer.toString(k);
```

```
            }

            //System.out.println("This is the day: " + temp);

          finalDate = temp + "/" + formattedMonth + "/" +  year;  //full    DD/MM/YYYY

          finalDatePalindrome = temp + formattedMonth + year;    //DDMMYYYY

           //System.out.println("This is final date: " + finalDate);

           checkPalindrome(finalDate, finalDatePalindrome);   // method call

           //System.out.println(temp);
           //System.out.println(formattedMonth);

           if (k==daysMonth && formattedMonth=="12")  // if it reached last day month of
December
           {
              year++;   // increase the year

              mmInt=0;  // this starts again new year.. 0 since used in zero index scenarios...

              newYearRollOver=true;  //sets flag new year

           }

           }
           processedFirstMonth=true;   // once flag set, no need to set back to false... since all
           // months across years will start from 1

        if (!newYearRollOver)  // this now deals with all other scenarios of increasing month
        //exception December above

        {
           mmInt=mmInt+1;
        }

        }  // this terminates the enum of traversing months

        monthcount++;

        }while(monthcount<=12);

             }  // end for loop (years)


     }  // terminates createdate method

}  // end of class
```