

MINI SOFTWARE PROJECT REPORT

CALENDAR GENERATION SYSTEM USING JAVA

Project Title: Calendar Generation System

Subject: Computer Applications

Programming Language: Java

Student Name: Neysa Mary Pramod

Class & Section: 9E

School Name: Ryan International School

Academic Year: 2020 - 2021

Certificate

This is to certify that **Neysa Mary Pramod of Class 9E** has successfully completed the mini software project titled “**Calendar Generation System Using Java**” as part of the Computer Applications curriculum during the academic year 2020-2021 under my guidance.

SUBMITTED TO:

MS. PAMELA MAITI

Acknowledgement

I sincerely thank my Computer teacher for guiding me throughout this project and helping me understand the concepts of Java programming. I also thank my parents and friends for their continuous support and encouragement during the completion of this project.

1. Objectives

The main objectives of this project are:

- To understand the basics of Java programming.
- To implement the concept of arrays and user-defined functions.
- To generate a calendar for any given month and year.
- To understand leap year calculation.
- To take user input using the Scanner class.
- To print formatted output in Java.

2. Introduction

A calendar is a very important tool used in daily life to organize days, dates, and events. In this project, a **Java-based calendar system** is developed which allows the user to enter a year, month name, and the weekday of the first day of the month. The program then generates and displays the complete calendar for that given month.

This project helps in understanding how real-life applications can be implemented using Java programming concepts.

3. Software Requirements

- **Operating System:** Windows / Linux / macOS
- **Programming Language:** Java
- **Compiler:** JDK (Java Development Kit)
- **IDE:** BlueJ / NetBeans / IntelliJ / Command Prompt

4. Description

The program performs the following tasks:

1. Accepts the **year** from the user.
2. Accepts the **month name** (e.g., January).
3. Accepts the **weekday of the first day** of that month.
4. Checks whether the entered year is a **leap year**.
5. Finds the **total number of days** in the given month.
6. Places all the dates properly in a **2D array**.
7. Prints the **formatted calendar** on the screen.

5. Program

```
import java.util.*;

class Calendar {

    int findMaxDay(String mname, int y) {

        String months[] = { "",
            "January",
            "February",
            "March",
            "April",
            "May",
            "June",
            "July",
            "August",
            "September",
            "October",
            "November",
            "December"
        };
    }
};
```

```
int D[] = {0,31,28,31,30,31,30,31,31,30,31,30,31};  
if ((y % 400 == 0) || ((y % 100 != 0) && (y % 4 == 0))) {  
    D[2] = 29;  
}
```

```
int max = 0;  
for (int i = 1; i <= 12; i++) {  
    if (mname.equalsIgnoreCase(months[i])) {  
        max = D[i];  
    }  
}  
return max;  
}
```

```
int findDayNo(String wname) {  
    String days[] = {  
        "Sunday",  
        "Monday",  
        "Tuesday",  
        "Wednesday",  
        "Thursday",
```

```
    "Friday",  
    "Saturday"  
};
```

```
int f = 0;  
for (int i = 0; i < 7; i++) {  
    if (wname.equalsIgnoreCase(days[i])) {  
        f = i;  
    }  
}  
return f;  
}  
void fillCalendar(int max, int f, String mname, int y) {  
    int A[][] = new int[6][7];  
    int x = 1, z = f;  
  
    for (int i = 0; i < 6; i++) {  
        for (int j = f; j < 7; j++) {  
            if (x <= max) {  
                A[i][j] = x;  
                x++;  
            }  
        }  
    }  
}
```



```
    }  
  }  
  f = 0;  
}
```

```
for (int j = 0; j < z; j++) {  
    A[0][j] = A[5][j];  
}
```

```
    printCalendar(A, mname, y);  
}
```

```
// Function for printing the calendar
```

```
void printCalendar(int A[][], String mname, int y) {
```

```
    System.out.println("\n\t");
```

```
    System.out.println("\t\t\t " + mname + " " + y);
```

```
    System.out.println("\t");
```

```
    System.out.println("\tSUN\tMON\tTUE\tWED\tTHU\tFRI\tSAT  
");
```

```
    System.out.println("\t");
```

```
for (int i = 0; i < 5; i++) {  
    for (int j = 0; j < 7; j++) {  
        if (A[i][j] != 0) {  
            System.out.print("\t " + A[i][j]);  
        } else {  
            System.out.print("\t ");  
        }  
    }  
    System.out.println("\n\t");  
}  
}
```

```
public static void main(String args[]) {  
    Calendar ob = new Calendar();  
    Scanner sc = new Scanner(System.in);  
  
    System.out.print("Enter the year : ");  
    int y = sc.nextInt();  
  
    System.out.print("Enter the month name (e.g. January) : ");  
    String mname = sc.next();  
}
```

```
        System.out.print("Enter the week day name (e.g. Sunday)  
of 1st day of "
```

```
        + mname + " : ");
```

```
String wname = sc.next();
```

```
int max = ob.findMaxDay(mname, y);
```

```
int f = ob.findDayNo(wname);
```

```
ob.fillCalendar(max, f, mname, y);
```

```
sc.close();
```

```
}
```

```
}
```

6. Output

Enter the year: 2025

Enter the month name: January

Enter the weekday of 1st day: Wednesday

January 2025

SUN	MON	TUE	WED	THU	FRI	SAT
	1	2	3	4		
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

7. Explanation of Functions Used

1. findMaxDay(String mname, int y)

- Finds the **maximum number of days** in the given month.
- Checks for leap year and assigns **29 days to February** if required.

2. findDayNo(String wname)

- Converts the **weekday name** (like Sunday, Monday) into its corresponding number (0–6).

3. fillCalendar(int max, int f, String mname, int y)

- Stores all date values in a **2D array (6×7)**.
- Adjusts the positions based on the starting weekday.

4. printCalendar(int A[][], String mname, int y)

- Prints the calendar in a **tabular format** with proper headings.

5. main() Method

- Takes all user inputs.
- Calls the above functions and displays the final output.

8. Advantages of the Project

- Easy to use.
- Fast and accurate.
- Helps in learning Java basics.
- Useful for date planning and scheduling.
- Demonstrates real-world application of arrays and conditions.

9. Limitations

- Works only for one month at a time.
- Requires correct spelling of month and weekday.
- Does not include holidays.
- Output is in text format only.

10. Applications

- Academic learning.
- Calendar generation tools.
- Scheduling systems.
- Desktop applications.

Conclusion

This project successfully demonstrates how a **calendar can be generated using Java programming**. Through this project, I learned about **functions, arrays, conditions, loops, and formatted output**. It improved my logical thinking and problem-solving skills. This project will help me in building more advanced Java applications in the future.