

## MIDTERM PROGRAMMING EXERCISE - EVENODD.JAVA

The screenshot displays the Visual Studio Code interface with the 'EvenOdd.java' file open. The editor shows the following code:

```
1 package act1;
2
3 import java.util.Scanner;
4
5 public class EvenOdd {
6     public static void main(String[] args) {
7         var sc = new Scanner(System.in);
8         System.out.print("Input number: ");
9         int input = sc.nextInt();
10        sc.close();
11
12        if(input % 2 == 0) System.out.println(input + " is Even number.");
13        else System.out.println(input + " is Odd number.");
14    }
15 }
```

The 'RUN' and 'DEBUG' buttons are visible in the top left. The 'TERMINAL' panel at the bottom shows the command prompt output:

```
PS D:\SDF_MIDTERM> & 'c:\Users\Gernel Esguerra\.vscode\extensions\vscjava.vscode-java-debug-0.29.0\scripts\launcher.bat' 'C:\Program Files\Java\jdk-15\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:49926' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-Dfile.encoding=UTF-8' '-cp' 'C:\Users\Gernel Esguerra\AppData\Roaming\Code\User\workspaceStorage\5a6b77c67ffbb0801374a2093c2a2ade\redhat.java\jdt_ws\SDF_MIDTERM_1cdb7851\bin' 'act1.EvenOdd'
Input number: 5
5 is Odd number.
PS D:\SDF_MIDTERM>
```

The status bar at the bottom indicates the current position is 'Ln 10, Col 20' with 'Spaces: 4', 'UTF-8', 'CRLF', 'Java', and 'JavaSE-15'.

## MIDTERM PROGRAMMING EXERCISE - ASCENDINGANDDESCENDING.JAVA

The screenshot displays the Visual Studio Code interface with the following components:

- Explorer Panel:** Shows a project structure with folders `act1` through `act10` and a `MAVEN` section. The file `AscendingandDescending.java` is selected.
- Editor Panel:** Contains the source code for `AscendingandDescending.java`. The code implements a sorting algorithm that first sorts an array in ascending order and then prints it in descending order.
- Terminal Panel:** Shows the output of the program execution, including the command used to run the program and the resulting output.

```
4 import java.util.Scanner;
5 public class AscendingandDescending {
6     public static void main(String[] args) {
7         var sc = new Scanner(System.in);
8         int[] num = new int[3];
9         for(int i = 0; i <= 2; i++){
10             System.out.print("Input a number: ");
11             num[i] = sc.nextInt();
12         }
13         sc.close();
14
15         int temp = 0;
16         for(int i = 0; i < num.length; i++){
17             for(int j = i; j < num.length; j++){
18                 if(num[i] > num[j]){
19                     temp = num[i];
20                     num[i] = num[j];
21                     num[j] = temp;
22                 }
23             }
24         }
25     }
26 }
```

Files\Java\jdk-15\bin\java.exe' '-agentlib:jdwp=transport=dt\_socket,server=n,suspend=y,address=localhost:49935' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-Dfile.encoding=UTF-8' '-cp' 'C:\Users\Gerne1 Esguerra\AppData\Roaming\Code\User\workspaceStorage\5a6b77c67ffbb0801374a2093c2a2ade\redhat.java\jdt\_ws\SDF\_MIDTERM\_1cdb7851\bin' 'act2.AscendingandDescending'

Input a number: 5  
Input a number: 9  
Input a number: 1  
Ascending: 1 5 9  
Descending: 9 5 1  
PS D:\SDF\_MIDTERM>

Ln 15, Col 22 Spaces: 4 UTF-8 CRLF Java JavaSE-15 12:09 PM

## MIDTERM PROGRAMMING EXERCISE – CONDOSALES.JAVA

The screenshot shows the Visual Studio Code editor with the `CondoSales.java` file open. The file is part of a project named `SDF_MIDTERM`, specifically under the `act3` directory. The code implements a simple menu-driven program for displaying condominium prices.

```
public static void main(String[] args) {  
    var sc = new Scanner(System.in);  
    var sb = new StringBuilder();  
    byte option = 0;  
    int price = 0;  
    sb.append("Please select view:\n");  
    sb.append("[1] Park view\n");  
    sb.append("[2] Golf Course view\n");  
    sb.append("[3] Lake view\n");  
    System.out.println(sb.toString());  
  
    System.out.print("Enter your option: ");  
    option = sc.nextByte();  
    sc.close();  
  
    switch(option){  
        case 1:  
            price = 150000;  
            break;  
        case 2:  
            price = 170000;  
            break;  
    }  
}
```

The terminal output shows the execution of the program:

```
a\jdt_ws\SDF_MIDTERM_1cdb7851\bin' act3.CondoSales'  
Please select view:  
[1] Park view  
[2] Golf Course view  
[3] Lake view  
  
Enter your option: 1  
Condominium Price is: $150000  
PS D:\SDF_MIDTERM>
```

The status bar at the bottom indicates the current position is Line 37, Column 6, with 4 spaces, using UTF-8 encoding and CRLF line endings. The Java version is JavaSE-15.

## MIDTERM PROGRAMMING EXERCISE - CONDOSALES2.JAVA

The screenshot displays the Visual Studio Code interface with the `CondoSales2.java` file open. The Explorer sidebar on the left shows a project structure with folders `act1` through `act10` and a `MAVEN` section. The main editor area shows the following Java code:

```
39 sb.setLength(0);
40
41 if(isValid){
42     sb.append("Select Garage Option: \n");
43     sb.append("[1] Garage\n");
44     sb.append("[2] Parking Space\n");
45     System.out.println(sb.toString());
46     sc.nextLine();
47     System.out.print("Enter your option: ");
48     option = sc.nextByte();
49     if(option == 1) price += 5000;
50     else if(option == 2);
51     else validFlag = false;
52 }
53 sc.close();
54 sb.setLength(0);
55 sb.append("Condominium Price is: $" + price + "\n");
56 if(validFlag == false && isValid == true){
57     sb.append("Parking option Invalid\n");
58 }
59 System.out.println(sb.toString());
60 }
61 }
```

The TERMINAL panel at the bottom shows the execution output:

```
Enter your option: 1
Select Garage Option:
[1] Garage
[2] Parking Space

Enter your option: 1
Condominium Price is: $155000

PS D:\SDF_MIDTERM>
```

The status bar at the bottom indicates the current position is Line 46, Column 53, with 4 spaces, UTF-8 encoding, and CRLF line endings. The system clock shows 12:13 PM.

## MIDTERM PROGRAMMING EXERCISE - CELLPHONESERVICE.JAVA

The screenshot shows the Visual Studio Code editor with the `CellPhoneService.java` file open. The file is located in the `act4` directory under the `SDF_MIDTERM` project. The code defines a `main` method that takes an array of strings as input and prints a recommendation based on the input values.

```
public static void main(String[] args) {
    var sc = new Scanner(System.in);
    String[] criteria = {"talk minutes: ", "text message sent: ", "gigabytes date used: "};
    int[] consumes = new int[3];
    for(byte i = 0; i < 3; i++){
        System.out.print("Maximum " + criteria[i]);
        consumes[i] = sc.nextInt();
        sc.nextLine();
    }
    sc.close();

    var sb = new StringBuilder();
    sb.append("\nRecommendation:\n");
    if(consumes[0] < 500 && consumes[1] <= 0 && consumes[2] <= 0) sb.append("Plan A at $49 per month\n");
    else if(consumes[0] < 500 && consumes[1] > 0 && consumes[2] <= 0) sb.append("Plan B at $55 per month\n");
    else if(consumes[0] >= 500 && consumes[1] <= 100 && consumes[2] <= 0) sb.append("Plan C at $61 per month\n");
    else if(consumes[0] >= 500 && consumes[1] >= 100 && consumes[2] <= 0) sb.append("Plan D at $70 per month\n");
    else if(consumes[0] >= 0 && consumes[1] >= 0 && consumes[2] > 0 && consumes[2] <= 2) sb.append("Plan E at $79 per m");
    else if(consumes[0] >= 0 && consumes[1] >= 0 && consumes[2] > 0 && consumes[2] >= 2) sb.append("Plan F at $87 per m");

    System.out.println(sb.toString());
}
```

The terminal output shows the execution of the program. It prompts for the maximum talk minutes, text message sent, and gigabytes date used. The user enters 500, 20, and 2 respectively. The program then recommends Plan E at \$79 per month.

```
a:\jdt_ws\SDF_MIDTERM_1cdb7851\bin' 'act4.CellPhoneService'
Maximum talk minutes: 500
Maximum text message sent: 20
Maximum gigabytes date used: 2

Recommendation:
Plan E at $79 per month

PS D:\SDF_MIDTERM>
```

## MIDTERM PROGRAMMING EXERCISE – PASTPRESENTFUTURE.JAVA

The screenshot shows the Visual Studio Code interface with the file `PastPresentFuture.java` open. The file is located in the `SDF_MIDTERM` project, under the `act5` folder. The code defines a `PastPresentFuture` class with a `main` method that takes an array of strings as input and prints the date and time based on the input criteria.

```
public class PastPresentFuture {  
    public static void main(String[] args) throws ParseException {  
        var sc = new Scanner(System.in);  
        String[] criteria = {"Month: ", "Day: ", "Year: "};  
        int[] values = new int[3];  
        for(byte i = 0; i < 3; i++){  
            System.out.print(criteria[i]);  
            values[i] = sc.nextInt();  
            sc.nextLine();  
        }  
        sc.close();  
  
        Date dateNow = new Date();  
        Calendar calendar = Calendar.getInstance(TimeZone.getTimeZone("Asia/Singapore"));  
        calendar.setTime(dateNow);  
        if(values[2] != calendar.get(Calendar.YEAR)) System.out.println("Date is not this year.");  
        else if(values[0] < calendar.get(Calendar.MONTH) +1) System.out.println("Date is earlier month this year.");  
        else if(values[0] > calendar.get(Calendar.MONTH) +1) System.out.println("Date is later month this year.");  
        else System.out.println("Date is this month.");  
    }  
}
```

The terminal output shows the execution of the program, displaying the date and time based on the input criteria:

```
PS D:\SDF_MIDTERM> d:; cd 'd:\SDF_MIDTERM'; & 'c:\Users\Gernel Esguerra\.vscode\extensions\vscjava.vscode-java-debug-0.29.0\scripts\launcher.bat' 'C:\Program Files\Java\jdk-15\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:49979' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-Dfile.encoding=UTF-8' '-cp' 'C:\Users\Gernel Esguerra\AppData\Roaming\Code\User\workspaceStorage\5a6b77c67ffbb0801374a2093c2a2ade\redhat.java\jdt_ws\SDF_MIDTERM_1cdb7851\bin' 'act5.PastPresentFuture'  
Month: 10  
Day: 2  
Year: 2020  
Date is earlier month this year.  
PS D:\SDF_MIDTERM>
```

## MIDTERM PROGRAMMING EXERCISE – PASTPRESENTFUTURE2.JAVA

The screenshot shows the Visual Studio Code editor with the file `PastPresentFuture2.java` open. The file is part of a project named `SDF_MIDTERM`. The code in the file is as follows:

```
13 for(byte i = 0; i < 3; i++){
14     System.out.print(criteria[i]);
15     values[i] = sc.nextInt();
16     sc.nextLine();
17 }
18
19 sc.close();
20
21 SimpleDateFormat formatter = new SimpleDateFormat("dd-MM-yyyy");
22 //Parsing the given String to Date object
23 Date date = new Date();
24 try {
25     date = formatter.parse(values[1] + "-" + values[0] + "-" + values[2]);
26 } catch (ParseException e) {
27     e.printStackTrace();
28 }
29
30 System.out.println();
31 if(date.before(formatter.parse(formatter.format(new Date())))) System.out.println(values[1] + "-" + values[0] + "-");
32 else if(date.after(formatter.parse(formatter.format(new Date())))) System.out.println(values[1] + "-" + values[0] + "-");
33 else System.out.println(values[1] + "-" + values[0] + "-" + values[2] + " is present.");
34 }
```

The terminal output shows the execution of the program:

```
Files\Java\jdk-15\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:49995' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-Dfile.encoding=UTF-8' '-cp' 'C:\Users\Gerne1 Esguerra\AppData\Roaming\Code\User\workspaceStorage\5a6b77c67ffbb0801374a2093c2a2ade\redhat.java\jdt_ws\SDF_MIDTERM_1cdb7851\bin' 'act5.PastPresentFuture2'
Month: 1
Day: 5
Year: 2021

5-1-2021 is future.
PS D:\SDF_MIDTERM>
```

## MIDTERM PROGRAMMING EXERCISE – PAY.JAVA

The screenshot shows the Visual Studio Code interface with the following components:

- Explorer:** Displays the project structure under 'SDF\_MIDTERM'. The 'act6' folder is expanded, showing 'Pay.java' as the active file.
- Editor:** Displays the code for 'Pay.java'. The code calculates net pay based on hours worked and a rate, with overtime pay for hours exceeding 40.
- Terminal:** Shows the output of running the program. It displays the date 'Month: 1, Day: 5, Year: 2021' and the message '5-1-2021 is future.'.

```
var sb = new StringBuilder();
float rate = 0, deductions = 0, reggrosspay = 0, ot = 0, netpay;
byte opt = 0, opt2;
int hrs = 0;
System.out.print("Enter skill level[1-3]: ");
opt = sc.nextByte();
sc.nextLine();

System.out.print("Enter attended hours: ");
hrs = sc.nextInt();
sc.nextLine();

rate = getRate(opt);
if(hrs <= 40){
    reggrosspay = hrs * rate;
    netpay = reggrosspay;
}
else{
    if((hrs / 40) > 1){
        ot = (((hrs / 40.0f) * 40.0f) - 40.0f) * (rate + (rate / 2));
        reggrosspay = 40.0f * rate;
        netpay = reggrosspay + ot;
    }
}
```

Files\Java\jdk-15\bin\java.exe' '-agentlib:jdwp=transport=dt\_socket,server=n,suspend=y,address=localhost:49995' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-Dfile.encoding=UTF-8' '-cp' 'C:\Users\Gerne1 Esguerra\AppData\Roaming\Code\User\workspaceStorage\5a6b77c67ffbb0801374a2093c2a2ade\redhat.java\jdt\_ws\SDF\_MIDTERM\_1cdb7851\bin' 'act5.PastPresentFuture2'

Month: 1  
Day: 5  
Year: 2021

5-1-2021 is future.  
PS D:\SDF\_MIDTERM>



## MIDTERM PROGRAMMING EXERCISE - JOBAPPLICANT.JAVA

The screenshot shows the Visual Studio Code editor with the file `JobApplicant.java` open. The Explorer sidebar on the left shows a project structure with folders `act1` through `act10` and a `MAVEN` section. The `JobApplicant.java` file is selected in the Explorer and its content is displayed in the editor. The code defines a `JobApplicant` class with attributes `name`, `phone`, `wproc`, `ssheets`, `dbase`, and `graphics`. It includes a constructor and several getter methods. The terminal at the bottom shows the output of running the program, displaying the date `5-1-2021` and the message `5-1-2021 is future.`.

```
package act7;

public class JobApplicant {
    String name, phone;
    boolean wproc, ssheets, dbase, graphics;

    JobApplicant(String name, String phone, boolean wproc, boolean ssheets, boolean dbase, boolean graphics){
        this.name = name;
        this.phone = phone;
        this.wproc = wproc;
        this.ssheets = ssheets;
        this.dbase = dbase;
        this.graphics = graphics;
    }

    public String getName(){return name;}
    public String getPhone(){return this.phone;}
    public boolean getWproc(){return this.wproc;}
    public boolean getSsheets(){return this.ssheets;}
    public boolean getDbase(){return this.dbase;}
    public boolean getGraphics(){return this.graphics;}
}
```

Files\Java\jdk-15\bin\java.exe' '-agentlib:jdwp=transport=dt\_socket,server=n,suspend=y,address=localhost:49995' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-Dfile.encoding=UTF-8' '-cp' 'C:\Users\Gerne1 Esguerra\AppData\Roaming\Code\User\workspaceStorage\5a6b77c67ffbb0801374a2093c2a2ade\redhat.java\jdt\_ws\SDF\_MIDTERM\_1cdb7851\bin' 'act5.PastPresentFuture2'

Month: 1  
Day: 5  
Year: 2021

5-1-2021 is future.  
PS D:\SDF\_MIDTERM>

## MIDTERM PROGRAMMING EXERCISE - TESTJOBAPPLICANT.JAVA

The screenshot displays the Visual Studio Code interface with the `TestJobApplicant.java` file open. The Explorer sidebar on the left shows the project structure, including the `SDF_MIDTERM` folder and its subfolders `act1` through `act10`. The `TestJobApplicant.java` file is selected in the Explorer and the Editor.

The code in `TestJobApplicant.java` is as follows:

```
18 }
19
20     ShowResult(applicants);
21 }
22
23 static void ShowResult(JobApplicant[] applicants){
24     for (JobApplicant jobApplicant : applicants) {
25         var sb = new StringBuilder();
26         sb.append(String.format("\nApplicant Name   : %s\n", jobApplicant.getName()));
27         sb.append(String.format("Phone Number   : %s\n", jobApplicant.getPhone()));
28         sb.append(String.format("Skills:\n"));
29         sb.append(String.format("Word Processing : %s\n", jobApplicant.getWproc() == true ? "Yes" : "No"));
30         sb.append(String.format("Spreadsheets   : %s\n", jobApplicant.getSsheets() == true ? "Yes" : "No"));
31         sb.append(String.format("Database       : %s\n", jobApplicant.getDbase() == true ? "Yes" : "No"));
32         sb.append(String.format("Graphics       : %s\n", jobApplicant.getGraphics() == true ? "Yes" : "No"));
33
34         byte i = 0;
35         if(jobApplicant.getWproc() == true) i++;
36         if(jobApplicant.getSsheets() == true) i++;
37         if(jobApplicant.getDbase() == true) i++;
38         if(jobApplicant.getGraphics() == true) i++;
39         sb.append(String.format("Remarks       : %s\n", i >= 3 ? "Qualified" : "Not-Qualified"));
40     }
41     System.out.println(sb.toString());
42 }
```

The Terminal window at the bottom shows the output of the program:

```
Applicant Name   : Jayjay
Phone Number     : 9511265847
Skills:
Word Processing   : No
Spreadsheets     : No
Database         : Yes
Graphics         : Yes
Remarks         : Not-Qualified
```

The status bar at the bottom indicates the current position is Line 39, Column 53, with 4 spaces, UTF-8 encoding, and CRLF line endings. The Java version is JavaSE-15.

## MIDTERM PROGRAMMING EXERCISE - AUTOMOBILES.JAVA

Visual Studio Code interface showing the file `Automobiles.java` in the `SDF_MIDTERM` project. The Explorer sidebar on the left shows the project structure, including folders `act1` through `act9` and a `MAVEN` section. The main editor displays the code for `Automobiles.java`, which includes methods for setting and getting attributes like `Id`, `Year`, `Model`, `Make`, `Color`, and `MPG`. The Terminal at the bottom shows the command prompt `PS D:\SDF_MIDTERM>`.

```
14 }
15
16 public void setId(int id){this.Id = id;}
17 public void setYear(int yr){this.Year = yr;}
18 public void setModel(String model){this.Model = model;}
19 public void setMake(String make){this.Make = make;}
20 public void setColor(String color){this.Color= color;}
21 public void setMPG(float mpg){this.mpg = mpg;}
22
23 public int getId(){
24     if(this.Id >= 0 && this.Id <= 9999) return this.Id;
25     else return 0;
26 }
27 public int getYear(){
28     if(this.Year >= 2000 && this.Year <= 2017) return this.Year;
29     else return 0;
30 }
31 public String getModel(){return this.Model;}
32 public String getMake(){return this.Make;}
33 public String getColor(){return this.Color;}
34 public float getMPG(){
35     if(this.mpg >= 10 && this.mpg <= 60) return this.mpg;
36     else return 0;
```

2: Java Debug Console

PS D:\SDF\_MIDTERM>

## MIDTERM PROGRAMMING EXERCISE - TESTAUTOMOBILES.JAVA

The screenshot displays the Visual Studio Code interface with the `TestAutomobiles.java` file open. The code is as follows:

```
act8 > TestAutomobiles.java > TestAutomobiles > main(String[])
5      Automobile auto1 = new Automobile(3655, 2012, "Prius", "Toyota", "Black", 12.5f);
6      Automobile auto2 = new Automobile(1000, 2000, "Corona", "Toyota", "Red", 30);
7      Automobile auto3 = new Automobile(12000, 1998, "Corona", "Toyota", "Red", 30);
8
9      auto1.setColor("White"); // auto1 set color to white
10
11     auto2.setModel("Civic"); // auto2 change model to Civic
12     auto2.setMake("Honda"); // auto2 change make to Honda
13
14     auto3.setMPG(68); // auto3 change mpg to 68 from 30
15
16     show(auto1);
17     show(auto2);
18     show(auto3);
19
20
21
22     static void show(Automobile automobiles){
23         var sb = new StringBuilder();
24         sb.append("\nCar ID : " + automobiles.getId() + "\n");
25         sb.append("Make : " + automobiles.getMake() + "\n");
26         sb.append("Model : " + automobiles.getModel() + "\n");
27         sb.append("Year : " + automobiles.getYear() + "\n");
```

The output console shows the results of the `show` method calls:

```
Make : Honda
Model : Civic
Year : 2000
Color : Red
MPG : 30.0

Car ID : 0
Make : Toyota
```

## MIDTERM PROGRAMMING EXERCISE – APARTMENT.JAVA

Visual Studio Code interface showing the `Apartment.java` file in the `SDF_MIDTERM` project. The Explorer sidebar on the left lists the project structure, including files like `CondoSales.java`, `CellPhoneService.java`, `Pay.java`, `JobApplicant.java`, `TestJobApplicant.java`, `Automobiles.java`, `TestAutomobiles.java`, `Apartment.java`, `TestApartments.java`, and `TwelveDays.java`. The `MAVEN` section indicates "No Maven project found".

The `Apartment.java` file content is as follows:

```
1 package act9;
2
3 public class Apartment {
4     private int Number;
5     private int BedCount;
6     private int BathCount;
7     private float Rent;
8
9     Apartment(int number, int beds, int baths, float rent){
10         this.Number = number;
11         this.BedCount = beds;
12         this.BathCount = baths;
13         this.Rent = rent;
14     }
15
16     public int getNumber(){return this.Number;}
17     public int getBedCount(){return this.BedCount;}
18     public int getBathCount(){return this.BathCount;}
19     public float getRent(){return this.Rent;}
20 }
21
```

The bottom panel shows the `TERMINAL` tab with the following output:

```
Make      : Honda
Model     : Civic
Year      : 2000
Color     : Red
MPG       : 30.0

Car ID    : 0
Make      : Toyota
```

The status bar at the bottom indicates the current position is `Ln 19, Col 45`, with `Spaces: 4`, `UTF-8` encoding, `CRLF` line endings, and the `Java` language. The system clock shows `12:24 PM`.

## MIDTERM PROGRAMMING EXERCISE – TESTAPARTMENTS.JAVA

The screenshot displays the Visual Studio Code interface with the file `TestApartments.java` open. The Explorer sidebar on the left shows a project structure with folders `SDF_MIDTERM` and `MAVEN`. The `SDF_MIDTERM` folder contains several Java files, including `TestApartments.java`, which is currently selected. The code in `TestApartments.java` is as follows:

```
16  var sc = new Scanner(System.in);
17  String[] cat = {"Minimum number of Bedrooms: ", "Minimum number of Baths: ", "Maximum rent willing to pay: "};
18  float[] values = new float[3];
19  for(int i = 0; i < 3; i++){
20      System.out.print(cat[i]);
21      values[i] = sc.nextFloat();
22      sc.nextLine();
23  }
24  sc.close();
25
26  showAvailable(apartment, values);
27  }
28
29  static void showAvailable(Apartment[] apartment, float[] values){
30      for (Apartment appart : apartment) {
31          var sb = new StringBuilder();
32          if((appart.getBedCount() <= (int)values[0] && appart.getBathCount() <= (int)values[1]) && appart.getRent() <= v
33              sb.append(String.format("Room Number      : %d\n", appart.getNumber()));
34              sb.append(String.format("Bedrooms       : %d\n", appart.getBedCount()));
35              sb.append(String.format("Bathrooms      : %d\n", appart.getBathCount()));
36              sb.append(String.format("Rent           : $%.2f\n", appart.getRent()));
37              System.out.println(sb.toString());
```

The bottom of the interface shows the `TERMINAL` tab with the following output:

```
Make      : Honda
Model     : Civic
Year      : 2000
Color     : Red
MPG       : 30.0

Car ID    : 0
Make      : Toyota
```

The status bar at the bottom indicates the current position is Line 35, Column 57, with 4 spaces, UTF-8 encoding, and CRLF line endings. The language is set to Java, and the JavaSE-15 runtime is selected.

## MIDTERM PROGRAMMING EXERCISE - TWELVEDAYS.JAVA

The screenshot shows the Visual Studio Code interface with the file `TwelveDays.java` open. The Explorer sidebar on the left shows a project named `SDF_MIDTERM` with various Java files. The main editor displays the code for `TwelveDays.java`, which includes a `main` method that takes a `String[]` argument and prints the lyrics of the Twelve Days of Christmas for a given day. The code uses a `switch` statement to handle days 1 through 12, with a `default` case for days 1 and 2.

```
act10 > TwelveDays.java > TwelveDays > main(String[])
22     str = String.format("On the %drd day of Christmas, my true love sent to me", dayofXmas);
23     break;
24     default:
25     str = String.format("On the %dth day of Christmas, my true love sent to me", dayofXmas);
26     break;
27 }
28
29 System.out.println(str);
30 switch (dayofXmas) {
31     case 12:
32         System.out.println("12 drummers drumming");
33     case 11:
34         System.out.println("11 pipers piping");
35     case 10:
36         System.out.println("10 lords-a-leaping");
37     case 9:
38         System.out.println("9 ladies dancing");
39     case 8:
40         System.out.println("8 maids-a-milking");
41     case 7:
42         System.out.println("7 swans-a-swimming");
43     case 6:
44         System.out.println("6 geese-a-laying");
45     case 5:
46         System.out.println("5 golden rings");
47     case 4:
48         System.out.println("4 calling birds");
49     case 3:
50         System.out.println("3 french hens");
51     case 2:
52         System.out.println("2 turtle doves");
53     case 1:
54         System.out.println("And a partridge in a pear tree!");
55 }
```

The terminal at the bottom shows the execution of the program. It prompts the user to enter the day of Christmas (1-12), and the output shows the lyrics for the 5th day of Christmas.

```
a\jdt_ws\SDF_MIDTERM_1cdb7851\bin' 'act10.TwelveDays'
Enter days of christmas [1-12]: 5
On the 5th day of Christmas, my true love sent to me
5 golden rings
4 calling birds
3 french hens
2 turtle doves
And a partridge in a pear tree!
PS D:\SDF_MIDTERM>
```

The status bar at the bottom indicates the current line and column (Ln 12, Col 25), the number of spaces (4), the encoding (UTF-8), the line ending (CRLF), and the Java version (JavaSE-15).