

# Viva 1 Occ7

**Group Name:** KodeLib

**Group Members:**

Full Name	Matric Number
YASEEN AYATULLAH KHAN	24216496/1
CHONG MING ZHE	25005796/1
MALAK MAHMOUD ABDELAAL	24226276/1
ABDUL RIZWAN MOHAMMED	24237402/1
IRSALINA DAMIA BINTI IBARIHIM	24003211/2
NURIN DAMIA BINTI MOHD SYAHRUL	24001335/2

**Question 1**

Done by: Abdul Rizwan Mohammed (24237402/1)

**1. Problem**

The task is to calculate the total library fine based on the number of overdue days, book code, category, and number of late returns. It validates inputs, extracts values manually without using advanced string methods, and calculates fines according to the rules for different book codes, additional penalties and discount conditions.

**2. Solution**

The code prompts the user to enter the number of cases for which fines are calculated. if the user enters 0 or less than 0, the user has to enter a number again. Then after that the system prompts the user to enter due days, book code, category and late returns in a single line. The system then extracts the due days, book code, category and late returns and checks if the value entered is valid or not. If not, the user is asked to enter a valid value. Once the values entered are valid, the program calculate the total fine according to the rules and display it. This process repeats until all cases have been processed.

**3. Sample Input & Output**

Sample input and output 1:

```
5
10 G S 3
--- Case 1 ---
Total fine: RM 16.50

0 R S 0
--- Case 2 ---
Total fine: RM 0.00

-1 C T 1
invalid input, try again
1 c t 3
--- Case 3 ---
Total fine: RM 9.60

70 C S 3
--- Case 4 ---
Total fine: RM 355.00

70 C S 0
--- Case 5 ---
Total fine: RM 345.00
```

Sample input and output 2:

```
3
3 T S 2
--- Case 1 ---
Total fine: RM 30.00

1 C K 2
invalid input, try again
1 C T 2
--- Case 2 ---
Total fine: RM 1.60

4 L S 0
invalid input, try again
4 m s -2
invalid input, try again
4 m s 0
--- Case 3 ---
Total fine: RM 0.80
```

#### 4. Source Code

```
package um.myum.viva1;

import java.util.Scanner;

public class Encik_Hafizs_Library_Fine_System {

    public static void main(String[] args) {

        Scanner inp = new Scanner(System.in);

        int cases=inp.nextInt();

        while(true){

            if(cases>0)

                break;

            else {

                System.out.println("Invalid input, try again");

                cases = inp.nextInt();

            }

        }

        inp.nextLine();
```

```
for (int case_no = 1; case_no <= cases; case_no++) {

    String str = inp.nextLine();

    double fine = 0;

    int late = 0;

    int due = 0;

    String code = "";

    String cat = "";

    int lent = str.length();

    for(int i = 0; i < 2; i++) {

        char u = str.charAt(i);

        if(u != ' ') {

            due = due * 10 + (u - '0');

        }

    }

    for(int i = lent-2; i < lent; i++) {

        char u = str.charAt(i);

        if(u != ' ') {

            late = late * 10 + (u - '0');

        }

    }

    for(int i = lent - 6; i < lent - 4; i++) {

        char u = str.charAt(i);

        if(u != ' ')

            code = u + "";

    }

}
```

```
}

for(int i = lent - 4; i < lent - 2; i++) {

    char u = str.charAt(i);

    if(u != ' ')

        cat = u + "";

}

while(true) {

    if((((lent<=9) && code.equalsIgnoreCase("R")) ||

        (code.equalsIgnoreCase("G")) ||

        (code.equalsIgnoreCase("M")) ||

        (code.equalsIgnoreCase("C")) ||

        (code.equalsIgnoreCase("T")))) &&

        ((cat.equalsIgnoreCase("S")) ||

        (cat.equalsIgnoreCase("T")))) &&

        ((due>=0)&&(late>=0)))

        break;

    else {

        System.out.println("invalid input, try again");

        str = inp.nextLine();

        lent = str.length();

        late = 0;

        due = 0;

        code = "";

        cat = "";

    }

}
```

```
    for(int i = 0; i < 2; i++) {  
        char u = str.charAt(i);  
        if(u != ' ') {  
            due = due * 10 + (u - '0');  
        }  
    }  
  
    for(int i = lent - 2; i < lent; i++) {  
        char u = str.charAt(i);  
        if(u != ' ') {  
            late = late * 10 + (u - '0');  
        }  
    }  
  
    for(int i = lent - 6; i < lent - 4; i++) {  
        char u = str.charAt(i);  
        if(u != ' ')  
            code = u + "";  
    }  
  
    for(int i = lent - 4; i < lent - 2; i++) {  
        char u = str.charAt(i);  
        if(u != ' ')  
            cat = u + "";  
    }  
}
```

```
if(code.equalsIgnoreCase("R") && due > 0)

    fine += 100;

else if(code.equalsIgnoreCase("G")) {

    if(due <= 7)

        fine = due * 0.50;

    else if(due <= 30)

        fine = (7 * 0.50) + ((due - 7) * 1.00);

    else

        fine = (7 * 0.50) + (23 * 1.00) + ((due - 30) * 2.00);

}

else if(code.equalsIgnoreCase("M"))

    fine = due * 0.20;

else if(code.equalsIgnoreCase("C")) {

    for (int day = 1; day <= due; day++) {

        if(day <= 10)

            fine += 2.00;

        else

            fine += 5;

    }

}

else {

    fine = due * 10;

    if(due > 15)

        fine += 200;
```

```
    }

    if(due > 60)

        fine += 25;

    if(late >= 3)

        fine += 10;

    if(cat.equalsIgnoreCase("T"))

        fine -= 0.20 * fine;

    else if(late==0 && due<=3)

        fine -= 0.50 * fine;

    System.out.println("--- Case " + case_no + " ---");

    System.out.printf("Total fine: RM %.2f\n", fine);

    System.out.println();

}

}

}
```



## **Question 2**

Done by: Chong Ming Zhe (25005796/1)

### **1. Problem**

While developing the “Kopi-Satu” Café, there are several problems encountered. One is the system need to accept multiple items with different prices and quantities from the customers without any errors. Next, the system have to calculate discounts, taxes, and cashback rewards based on total amount, day of the week, time and membership. Lastly, the output of receipt have to be organized and clear, showing all items and the accurate total amount.

### **2. Solution**

To overcome these issues, the system can use loops and array to handle multiple items and prices efficiently. The system will prompt the user to enter valid input to make sure there is no any error in calculation. Besides, conditional statement can be used in calculating discounts, taxes and cashback rewards correctly based on respective input. Calculation need to be done step by step to avoid errors and the output of receipt have to be in 2 decimal places by using printf statement.

### **3. Sample Input & Output**

```
run:
Enter item price (0 to finish): 15.00
Enter item price (0 to finish): 22.00
Enter item price (0 to finish): 0
Enter day of week: Friday
Enter hour (24-hour format): 16
Is customer a member (Y/N)? Y
---- Kopi-Satu Receipt -----
Subtotal:                      RM 37.00
Service Tax (8%):              RM 2.96
Total before discount:         RM 39.96
Student Discount (10%):        RM 4.00
Happy Hour Discount (5%):      RM 2.00
-----
Total Payable:                 RM 33.97
Loyalty Cashback (2%):         RM 0.68
-----
Final Amount to Collect:       RM 33.97
BUILD SUCCESSFUL (total time: 30 seconds)
```

```
run:
Enter item price (0 to finish): 4
Enter item price (0 to finish): 2
Enter item price (0 to finish): -1
Invalid amount. Price cannot be negative. Please re-enter.
Enter item price (0 to finish): 0
Enter day of week: Saturday
Enter hour (24-hour format): 22
Is customer a member (Y/N)? N
---- Kopi-Satu Receipt -----
Subtotal:                      RM 6.00
Service Tax (6%):              RM 0.36
Total before discount:         RM 6.36
-----
Total Payable:                 RM 6.36
-----
Final Amount to Collect:      RM 6.36
BUILD SUCCESSFUL (total time: 22 seconds)
```

#### 4. Source Code

```
package billingsystem;

import java.util.Scanner;

public class BillingSystem {

    public static void main(String[] args) {

        Scanner input = new Scanner(System.in);

        double price;

        double subtotalPrice = 0;

        double serviceTax;

        //Prompt user to input item price

        while (true) {

            System.out.print("Enter item price (0 to finish): ");
```

```
price = input.nextDouble();

if (price < 0) {
    System.out.println("Invalid amount. Price cannot be negative. Please re-
enter.");
    continue;
}

if (price == 0) break;

subtotalPrice += price;
}

//Prompt user to enter day of the week (NO ARRAY)
String day;
while (true) {
    System.out.print("Enter day of week: ");
    day = input.next();

    if (day.equalsIgnoreCase("Monday") ||
        day.equalsIgnoreCase("Tuesday") ||
        day.equalsIgnoreCase("Wednesday") ||
        day.equalsIgnoreCase("Thursday") ||
        day.equalsIgnoreCase("Friday") ||
        day.equalsIgnoreCase("Saturday") ||
        day.equalsIgnoreCase("Sunday")) {
```

```
        break; // valid
    }

    System.out.println("Invalid day. Please enter a valid day.");
}

//Prompt the user to enter hour in 24-hour format
int hour;
while (true) {
    System.out.print("Enter hour (24-hour format): ");
    hour = input.nextInt();
    if (hour >= 0 && hour <= 23) break;
    System.out.println("Invalid hour. Must be between 0 and 23.");
}

// --- Membership ---
System.out.print("Is customer a member (Y/N)? ");
char member = input.next().toUpperCase().charAt(0);

// --- Receipt ---
System.out.println("---- Kopi-Satu Receipt -----");
System.out.printf("Subtotal:          RM %.2f\n", subtotalPrice);
```

```
int service;

if (subtotalPrice <= 30) {

    serviceTax = subtotalPrice * 0.06;

    service = 6;

} else if (subtotalPrice <= 100) {

    serviceTax = subtotalPrice * 0.08;

    service = 8;

} else {

    serviceTax = subtotalPrice * 0.10;

    service = 10;

}

System.out.printf("Service Tax (%d%%):      RM %.2f\n", service, serviceTax);

double totalBeforeDiscount = subtotalPrice + serviceTax;

System.out.printf("Total before discount:  RM %.2f\n", totalBeforeDiscount);

boolean isWeekday =

    day.equalsIgnoreCase("Monday") ||

    day.equalsIgnoreCase("Tuesday") ||

    day.equalsIgnoreCase("Wednesday") ||

    day.equalsIgnoreCase("Thursday") ||

    day.equalsIgnoreCase("Friday");
```

```
boolean isWeekend = !isWeekday;

double discount1 = 0, discount2 = 0, discount3 = 0;

if (isWeekday && totalBeforeDiscount > 25) {
    discount1 = totalBeforeDiscount * 0.10;
    System.out.printf("Student Discount (10%%): RM %.2f\n", discount1);
}

if (isWeekday && hour >= 15 && hour < 17) {
    discount2 = totalBeforeDiscount * 0.05;
    System.out.printf("Happy Hour Discount (5%%): RM %.2f\n", discount2);
}

if (isWeekend && subtotalPrice >= 50) {
    discount3 = totalBeforeDiscount * 0.05;
    System.out.printf("Weekend Discount (5%%): RM %.2f\n", discount3);
}

double totalDiscount = discount1 + discount2 + discount3;

System.out.println("-----");

double total = totalBeforeDiscount - totalDiscount;

System.out.printf("Total Payable: RM %.2f\n", total);

if (member == 'Y') {
```

```
double cashback = total * 0.02;

System.out.printf("Loyalty Cashback (2%%):  RM %.2f\n", cashback);

}

System.out.println("-----");

System.out.printf("Final Amount to Collect:  RM %.2f\n", total);

}

}
```

**Question 3**

Done by: Malak Mahmoud Abdelaal (24226276/1)

**1. Problem**

Amir, one of the friends, is working on his Final Year Project (FYP), a social networking app for university students called “UniVerse.” At first, he only concentrated on password strength in the registration process, making sure users created secure passwords. But under the guidance of Dr. Hidayah, a cybersecurity expert, Amir realized that wasn’t enough.

The real problem is that UniVerse needs a two-tier validation system:

1. Username integrity – ensuring usernames are unique, appropriate, and resistant to misuse.
2. Password security – enforcing strong, complex passwords that can withstand attacks.

This shift shows Amir that building UniVerse isn’t just about functionality; it’s about protecting users from the start. The challenge is making registration both secure and user-friendly, without compromising safety.

**2. Solution**

To meet the new security standards set by Dr. Hidayah, Amir now has to redesign (UniVerse)’s registration module with a two-tier validation system. The first-tier checks username integrity—making sure usernames are clean, unique, and resistant to abuse. The second tier enforces password security, which goes beyond just complexity.

To solve this, Amir is writing a validation program that applies strict rules to both inputs:

- **Usernames** must follow format rules, avoid reserved words, and pass uniqueness checks.
- **Passwords** must meet multiple criteria: length, character variety, and resistance to common attacks.



Each password is then rated based on how many rules it satisfies:

Rules met	Password Strength
$\leq 3$	Weak
4-5	Moderate
6	Strong
7	Very Strong

This system forces users to follow the rules step by step, with Amir coding each check into the registration flow. It's not just about getting users signed up- it's about making sure they're protected from the moment they create their own account.

### 3. Sample Input & Output

#### Sample Input 1:

```
Enter username: alice
Enter password: aLicE2024!
```

#### Sample Output 1:

```
Password Strength: Strong
```

```
Enter username: alice
Enter password: aLicE2024!
Password Strength: Strong
BUILD SUCCESSFUL (total time: 43 seconds)
```

#### Sample Input 2:

```
Enter username: bob
Enter password: password
```

#### Sample Output 2:

```
Invalid username
```

```
Enter username: bob
Enter password: password
Invalid username
BUILD SUCCESSFUL (total time: 7 seconds)
```

**Sample Input 3:**

**Enter Username:** malak  
**Enter Password:** mAlAk014 <3

**Sample Output 3:**

**Password Strength:** Moderate

```
Enter username: malak
Enter password: mAlAk014<3
Password Strength: Moderate
BUILD SUCCESSFUL (total time: 39 seconds)
```

**4. Source Code**

```
import java.util.Scanner;

public class SecurityCheckp {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.print("Enter username: ");

        String username = sc.nextLine();

        System.out.print("Enter password: ");

        String password = sc.nextLine();

        // Step 1: Check username

        boolean validUsername = true;
```

```
// Rule 1: length between 5 and 15

if (username.length() < 5 || username.length() > 15) {

    validUsername = false;

}

// Rule 2: must start with lowercase letter

char first = username.charAt(0);

if (!Character.isLowerCase(first)) {

    validUsername = false;

}

// Rule 3: only lowercase, digits, underscore

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

    char c = username.charAt(i);

    if (!(Character.isLowerCase(c) || Character.isDigit(c) || c == '_')) {

        validUsername = false;

    }

}

if (!validUsername) {

    System.out.println("Invalid username");

    return; // stop program

}

// Step 2: Check password
```

```
int rulesMet = 0;

// Rule 1: length >= 8
if (password.length() >= 8) {
    rulesMet++;
}

boolean hasUpper = false;
boolean hasLower = false;
boolean hasDigit = false;
boolean hasSpecial = false;
boolean hasSpace = false;

String specialChars = "!@#$%^&*";

for (int i = 0; i < password.length(); i++) {
    char c = password.charAt(i);
    if (Character.isUpperCase(c)) hasUpper = true;
    if (Character.isLowerCase(c)) hasLower = true;
    if (Character.isDigit(c)) hasDigit = true;
    if (specialChars.indexOf(c) != -1) hasSpecial = true;
    if (c == ' ') hasSpace = true;
}

if (hasUpper) rulesMet++;
```

```
        if (hasLower) rulesMet++;

        if (hasDigit) rulesMet++;

        if (hasSpecial) rulesMet++;

        if (!hasSpace) rulesMet++;

        if (!password.toLowerCase().contains(username.toLowerCase())) rulesMet++;

    }

    // Step 3: Rating

    String rating;

    if (rulesMet <= 3) {

        rating = "Weak";

    } else if (rulesMet <= 5) {

        rating = "Moderate";

    } else if (rulesMet == 6) {

        rating = "Strong";

    } else {

        rating = "Very Strong";

    }

    System.out.println("Password Strength: " + rating);

}

}
```

**Question 4**

Done by: Irsalina Damia Binti Ibarihim (24003211/2)

**1. Problem**

During the Hari Merdeka, the Malaysian government decides to reward lucky citizens based on their MyKad (IC) numbers. Your mission is to analyse the IC number and determine whether a citizen qualifies as a “Lucky Winner.” The IC number contains certain data that will be decoded and evaluated according to the following criteria:

- Birthdate extraction. Where the first 2 digit represents the user year of birth year. if the first 2 digit is 00-25 it interpreted as 2000-2025 while 26-99 correspond to 1926-1999.
- Gender Identification. Where the last digit is determined by odd number represent male while even number represent female.
- Birth Month Classification. The birth month must be identified as either a long month or short month.
- Digit Sum Calculation. The system must compute the sum of all digit in the IC number.

Lucky winner. Using the gender, birth month and sum value. If male the total digit divided by 5 and short month is lucky otherwise not lucky. If female the total digit divided by 7 and a long month is lucky winner otherwise not lucky.

**2. Solution**

For user to enter the IC number with the dash. Use string input.NextLine(). To check whether user put the dash or not, the condition if else statement is used if true it will print error and system exit else the code is continued.

**1. Birthday extraction**

- The IC contains the birthdate in the first six digits (YYMMDD). The program extracts: Day characters at index 4-5, Month characters at index 2-3, Year characters at index 0-1. The year is interpreted using the rule: Years 00-25 = 2000-2025, Years 26-99 = 1926-1999. The program prints the birthdate in full: DD/MM/YYYY

**2. Gender Determination**

- The program reads the last digit of the IC (index 13). Using the rule: Odd Male, Even Female. it determines and displays the gender.

**3. Sum of All Digits**

- To compute the total of all digits in the IC, the program loops through every numeric character while skipping the hyphens. It performs three loops: From index 0 to 5 (first 6 digits), From index 7 to 8 (two digits after first hyphen), From index 10 to 13 (last four digits). Each character is converted to a numeric value and added to the running sum. The final digit sum is displayed.

#### 4. Month Classification

- The extracted birth month is classified as: Long Month 31 days (Jan, Mar, May, Jul, Aug, Oct, Dec). Short Month 28 or 30 days (Feb, Apr, Jun, Sep, Nov). A switch statement assigns the month type as either "long" or "short".

#### 5. Lucky Winner Evaluation

- The program applies gender-specific rules to determine luckiness: For Males, A male is considered Lucky if: Total digit sum is divisible by 5, AND Birth month is a Short Month. Otherwise, the male is Not Lucky. For Females, A female is considered Lucky if: Total digit sum is divisible by 7, AND Birth month is a Long Month. Otherwise, the female is Not Lucky. The program prints "yes" for lucky individuals and "no" for others.

#### 6. Summary of Output

- The program outputs the following information:
  - Full birthdate (DD/MM/YYYY)
  - Gender
  - Sum of all digits in the IC
  - Month classification (used internally)
  - Lucky Winner status: yes / no

### 3. Sample Input & Output

```
Enter IC number (YYMMDD-##-####) : 030831-12-0136
Birth date:31/08/2003
Gender : female
Sum of digit :28
Lucky winner : yes
BUILD SUCCESSFUL (total time: 17 seconds)
|
```

```
Enter IC number (YYMMDD-##-####) : 060721334356
Error.
BUILD SUCCESSFUL (total time: 18 seconds)
|
```

```
Enter IC number (YYMMDD-##-####) : 010203-04-0506
Birth date:03/02/2001
Gender : female
Sum of digit :21
Lucky winner : no
BUILD SUCCESSFUL (total time: 13 seconds)
|
```

#### 4. Source Code

```
package viva;

import java.util.Scanner;
public class Viva {

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.print("Enter IC number (YYMMDD-##-####) : ");
        String IC = input.nextLine();

        if (IC.charAt(6)!='-' || IC.charAt(9)!='-'){
            System.out.println("Error.");
            System.exit(0); // return
        }

        String date = IC.substring(4,6);
        String month = IC.substring(2,4);

        System.out.print("Birth date:" + date + "/" + month + "/");

        char year = IC.charAt(0);
        char year2 = IC.charAt(1);
        int yearr = Character.getNumericValue(year);
        int yearr2 = Character.getNumericValue(year2);

        if (yearr == 0 || yearr == 1) {
            System.out.println("20"+ year + year2);
        } else if ( yearr == 2 ) {
            if ( yearr2 > 1 && yearr2 <= 5){
                System.out.println("20"+ year + year2);
            } else {
                System.out.println("19"+year+year2);
            }
        } else {
            System.out.println("19"+year+year2);
        }
    }

    char gen = IC.charAt(13) ;
    int gender = Character.getNumericValue(gen);

    String gend ;
    if (gender%2 == 0) {
        gend = "female";
        System.out.println("Gender : female");
    } else {
```



```
        gend = "male";
        System.out.println("Gender : male");
    }

    int sum = 0;

    for(int i = 0 ; i < 6; i ++){
        char c = IC.charAt(i);
        int num = Character.getNumericValue(c);
        sum += num ;
    }
    for(int j = 7 ; j < 9 ; j++){
        char d = IC.charAt(j);
        int num = Character.getNumericValue(d);
        sum += num ;
    }
    for(int k = 10 ; k < 14; k++){
        char e = IC.charAt(k);
        int num = Character.getNumericValue(e);
        sum += num ;
    }
    System.out.println("Sum of digit :"+ sum);

    String m = "";

    switch (month) {
        case "01":
        case "03":
        case "05":
        case "07":
        case "08":
        case "10":
        case "12":
            m = "long";
            break;
        case "02":
        case "04":
        case "06":
        case "09":
        case "11":
            m = "short";
            break;
    }

    System.out.print("Lucky winner : ");
    if ( gend.equals("male")) {
        if ( m.equals("short") && sum%5 == 0 ) {
            System.out.println("yes");
        } else {
```

```
        System.out.println("no");
    }
}

if ( gend.equals("female")) {
    if ( m.equals("long") && sum%7 == 0 ) {
        System.out.println("yes");
    } else {
        System.out.println("no");
    }
}
}
}
```

**Question 5**

Done by: Nurin Damia Binti Mohd Syahrul (24001335/2)

**1. Problem**

In the kingdom of CIPHERIA, the secret agents constantly communicate with their troop but to prevent it to get leak to the neighbouring kingdom, CRYTONIA. They decided to use a special encoding system.

How it works:

- Convert each character in a sentence into its ASCII decimal value.
- Convert ASCII decimal value into an 8-bit binary string.
- Invert the binary string (0-->1 and 1 -->0).
- Convert the inverted binary string into decimal.

**2. Solution**

STEP 1 - Secret agents input the original message

- The user enters a word or a full sentence.
- The program read the entire string using `Scanner.nextLine()`. (This will also read the sentence that contains a space in it)

STEP 2 - Process each character in the message

A loop is used to make sure every character undergo all the the process separately:

1. Convert character to ASCII decimal

Using:

```
int ascii=(int) ch;
```

2. Convert ASCII to 8-bit binary

Using:

```
String binaryString = Integer.toBinaryString((int) ch);
```

```
String paddedBinary = String.format("%8s", binaryString).replace(' ', '0'); //make sure there's  
8 bits
```

### 3. Invert the binary

Using:

```
String invertedBinary = paddedBinary.replace('0', '#').replace('1', '0').replace('#', '1');
```

## is a temporarily place for 0, to make sure the output doesn't mess up and output all 1

### 4. Convert the inverted binary to decimal

Using:

```
int decimalNumber = Integer.parseInt(invertedBinary, 2);
```

### 5. Output the encoded message

- Every encoded decimal value is printed in one line
- Have spaces in between each character to prevent from mixing up the meaning behind the numbers

## 3. Sample Input & Output

Given:

---

```
run:
Enter Original Message: hello world
Encoded Message:
151 154 147 147 144 223 136 144 141 147 155 BUILD SUCCESSFUL (total time: 6 seconds)
|
```

Other examples:

---

```
run:
Enter Original Message: We Found Treasure!
Encoded Message:
168 154 223 185 144 138 145 155 223 171 141 154 158 140 138 141 154 222 BUILD SUCCESSFUL (total time: 18 seconds)
|
```

---

```
run:
Enter Original Message: !!! SEND HELP !!!
Encoded Message:
222 222 222 223 172 186 177 187 223 183 186 179 175 223 222 222 222 BUILD SUCCESSFUL (total time: 3 seconds)
|
```

```
run:
Enter Original Message: 1PM we attack
Encoded Message:
206 175 178 223 136 154 223 158 139 139 158 156 148 BUILD SUCCESSFUL (total time: 11 seconds)
```

#### 4. Source Code

```
package question.pkg5.viva;

import java.util.Scanner;

public class Question5Viva {

    public static void main(String[] args) {

        Scanner scanner=new Scanner(System.in);

        System.out.print("Enter Original Message: ");

        String word=scanner.nextLine();

        System.out.print("Encoded Message:\n");

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

            char ch = word.charAt(i); // returns the character at the specified index in a string.

            String binaryString = Integer.toBinaryString((int) ch);    //ASCII Decimal -> binary

            String paddedBinary = String.format("%8s", binaryString).replace(' ', '0');    //
making sure its 8-bits

            String invertedBinary = paddedBinary.replace('0', '#').replace('1', '0').replace('#',
'1');    // switching 1 to 0/ 0 to 1

            int decimalNumber = Integer.parseInt(invertedBinary, 2); //inverted back to decimal

            System.out.print(decimalNumber + " ");
```

```
}  
  
}  
  
}
```

**Question 6**

*Done by: Yaseen Ayatullah Khan (24216496/1)*

**1. Problem**

The user plays a small console-playable game titled 'Dragon Egg Quest'. There are a total of 10 chests, 3 of them have a dragon egg each, and 2 of the 10 chests are cursed (these are randomly generated). The user gets 10 attempts to correctly guess which 3 chests have the dragon eggs. If the user guesses a chest number which is cursed (even if it has a dragon egg), the user's attempts decrease by 2 instead of the usual decrement by 1.

The player/user wins the game if they enter all the correct chest numbers (ones that have dragon egg) before their attempts run out. Therefore, if the attempts run out and the user/player has not found all 3 of the eggs, they lose the game.

**2. Solution**

Firstly the libraries for input (Scanner) and random number generation (Random).

Then we would set them up by assigning the method of the library to a new object.

For initialisations, we need variable for the integer that the user enters as their guess, a variable to count how many eggs are found (initial 0), and how many attempts are left (initial 10).

In three variables used for chests that would have the eggs, we would use the Random generator object with bounds of 1 and 11.

In two variables we would do the same but these will be used to store the chest numbers for cursed chests.

A boolean variable will be used for the main while loop to work properly so that its value would be assigned to true when the user either manages to guess all 3 of the chests that have eggs, or the user's attempts run out (attempts = 0).

We would also need three variables to store three differences between the guessed integer and the three egg-containing chests so that we can find the nearest egg later (which is another variable we will initialise alongside the difference variables).

The main part of the whole program, a while loop, is then implemented with one condition that is whether the endLoop variable remains false or not (it runs until endLoop equals to true). The loop contains these parts:

Getting the guess as input, checking whether the guess matches the number of a cursed chest (if it does then we decrease attempts by 2, else attempts just decrements by 1), and checking whether the guess matches any chest with an egg and if it does then setting that particular chest variable's value to an extreme one.

(else case) If the guess does not match any egg-containing chest, then the difference variables are used obtain the difference between the guess and the number of each egg-containing chest. These are used to find out the smallest difference and then that chest's number is assigned to the variable named nearest. Then the three difference variables are checked to see if even one of them is smaller than or equal to 3, and if yes then the hint logic is implemented (if guess is bigger than nearest then ask to guess lower, and vice versa). Otherwise, the user is told that their guess is far away from any egg-containing chest. At the end of this big else-case, the user is told that there is no egg present in the chest they guessed so they should keep guessing.

Then the logic to continue or end the game is implemented: if the eggs found variable is already 3, or attempts is 0 then the endLoop boolean variable is set to true so that the main while loop cannot continue to run anymore.

### 3. Sample Input & Output

```
Welcome to the Dragon Egg Quest!
There are 10 chests, 3 dragon eggs, and 2 cursed chests.
You have 10 attempts to find all dragon eggs.

Guess a chest (1-10):
4
Warm! You're very close to a dragon egg!
Hint: Try a lower chest number.
No egg here, keep searching!
Attempts left: 9

Guess a chest (1-10):
1
You found a dragon egg!
Attempts left: 8

Guess a chest (1-10):
6
Warm! You're very close to a dragon egg!
Hint: Try a lower chest number.
No egg here, keep searching!
Attempts left: 7
```



```
Guess a chest (1-10):
5
You found a dragon egg!
Attempts left: 6

Guess a chest (1-10):
9
This chest is cursed! Beware!
You found a dragon egg!
Attempts left: 4

Congratulations! All dragon eggs are safe!
```

#### 4. Source Code

```
package dragon_egg_quest;

import java.util.Scanner;

import java.util.Random;

public class VivaQuestionSixYaseen {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        Random rand = new Random();

        // initialisations

        int guess, found = 0, attempts = 10;

        int eggChest1 = rand.nextInt(1,11), eggChest2 = rand.nextInt(1,11), eggChest3 =
rand.nextInt(1,11);

        int cursedChest1 = rand.nextInt(1,11), cursedChest2 = rand.nextInt(1,11);

        boolean guessedAll = false, endLoop = false;

        int diff1, diff2, diff3, nearest; //variables to check nearest remaining egg

        //game starts
```

```
System.out.println("Welcome to the Dragon Egg Quest!" + "\n" + "There are 10  
chests, 3 dragon eggs, and 2 cursed chests." + "\n" + "You have 10 attempts to find all  
dragon eggs." + "\n");
```

```
//main loop structure
```

```
while (endLoop == false) {
```

```
    System.out.println("Guess a chest (1-10): ");
```

```
    guess = scanner.nextInt();
```

```
//cursed chests and updating attempts
```

```
if (guess == cursedChest1 || guess == cursedChest2) {
```

```
    System.out.println("This chest is cursed! Beware!");
```

```
    attempts -= 2;
```

```
} else {
```

```
    attempts -= 1;
```

```
}
```

```
if (guess == eggChest1 || guess == eggChest2 || guess == eggChest3) {
```

```
    System.out.println("You found a dragon egg!");
```

```
    found++;
```

```
// assigning null/extreme values to chest found with egg
```

```
if (guess == eggChest1) {
```

```
    eggChest1 = 100;
```

```
} else if (guess == eggChest2) {
```

```
    eggChest2 = 100;
```

```
} else {
```

```
    eggChest3 = 100;
```

```
}
```

```
} else { //if the chest is empty

    diff1 = eggChest1 - guess;

    diff2 = eggChest2 - guess;

    diff3 = eggChest3 - guess;

    //finding nearest remaining egg

    if (diff1 < diff2 && diff1 < diff3) {

        nearest = eggChest1;

    } else if (diff2 < diff1 && diff2 < diff3) {

        nearest = eggChest2;

    } else {

        nearest = eggChest3;

    }

    //checking nearby chests

    if ((diff1) <= 3 || (diff2) <= 3 || (diff3) <= 3) {

        System.out.println("Warm! You're very close to a dragon egg!");

        //hints

        if (guess < nearest) {

            System.out.println("Hint: Try a higher chest number.");

        } else if (guess > nearest) {

            System.out.println("Hint: Try a lower chest number.");

        }

    } else {

        System.out.println("Cold! You're far from any dragon egg!");

    }

}
```

```
        System.out.println("No egg here, keep searching!");
    }

    System.out.println("Attempts left: " + attempts + "\n");

    //logic to continue or end the game
    if (found == 3) {

        System.out.println("Congratulations! All dragon eggs are safe!");

        guessedAll = true;

        endLoop = true;

    } else if (attempts == 0) {

        System.out.println("Game Over! Some dragon eggs remain hidden!");

        endLoop = true;

    }

}

}
```