SE Lab 7 202201477

Q.2 Debugging of code fragment

1. Armstrong number

- I have identified two errors. The calculation of remainder is wrong in this question.And the loop termination condition is also wrong.
- To fix this issue we should set our break point where the remainder is calculated and the number is updated after the loop Content executed.

Corrected Code:

```
class Armstrong{
    public static void main(String args[]){
        int num = Integer.parseInt(args[0]);
        int n = num; //use to check at last time
        int check=0,remainder;
        while(num > 0){
            remainder = num % 10;
            check = check + (int)Math.pow(remainder,3);
            num = num / 10;
        }
        if(check == n)
            System.out.println(n+" is an Armstrong Number");
        else
            System.out.println(n+" is not a Armstrong Number");
    }
}
```

2. LCM & GCD

- In GCD function there are two errors .
  1. a = (x > y) ? y : x;
  Here the condition should be like this :if x>y then a=x

  2. Loop termination condition
  The condition should be like this while(b !=0)

- In LCM function there is a logical error in loop termination Condition. We have to find the smallest number which is Divisible by both x and y.

Corrected code:

```
static int gcd(int x, int y)
{
    int r=0, a, b;
    a = (x > y) ? x : y; // a is greater number
    b = (x < y) ? x : y; // b is smaller number

    r = b;
    while(a % b != 0) {
        r = a % b;
        a = b;
        b = r;
    }
    return r;
}

static int lcm(int x, int y)
{
    int a;
    a = (x > y) ? x : y; // a is greater number
    while(true)
    {
        if(a % x == 0 && a % y == 0)
            return a;
        ++a;
    }
}
```

3. KnapSack

- First error in the code is in the line where we choose the option to not take the item. In this option1=opt[n-1][w]
- Second error is there in the condition where we have a condition If we can take an item or not. The condition should be like this (weight[n]<=w).
- Third error is when we take the current item the we add the profit of the current item and the max profit obtained from n-1 items and w-weight[n] capacity.

Corrected part of the Code :

```
int[][] opt = new int[N+1][W+1];
boolean[][] sol = new boolean[N+1][W+1];

for (int n = 1; n <= N; n++) {
    for (int w = 1; w <= W; w++) {

        // don't take item n
        int option1 = opt[n-1][w];

        // take item n
        int option2 = Integer.MIN_VALUE;
        if (weight[n] <= w) option2 = profit[n] + opt[n-1][w-weight[n]];

        // select better of two options
        opt[n][w] = Math.max(option1, option2);
        sol[n][w] = (option2 > option1);
    }
}
```

4.Magic Number

- The error is in the looping condition. while(sum !=0) this is the correct condition and also remainder calculation is wrong and There is a multiplication instead of addition also the increment

Condition of the number is also wrong and also at one place a semicolon is not present.

Corrected Code :

```java
import java.util.*;
public class MagicNumberCheck
{
    public static void main(String args[])
    {
        Scanner ob=new Scanner(System.in);
        System.out.println("Enter the number to be checked.");
        int n=ob.nextInt();
        int sum=0,num=n;
        while(num>9)
        {
            sum=num;int s=0;
            while(sum != 0)
            {
                s=s+(sum%10);
                sum=sum/10;
            }
            num=s;
        }
        if(num==1)
        {
            System.out.println(n+" is a Magic Number.");
        }
        else
        {
            System.out.println(n+" is not a Magic Number.");
        }
    }
}
```

5. Merge short:

- Here there is an error in mergesort() function where we divide An array into two halves. In this there are two functions

lefthalf and righthalf. we have to pass array as argument.but in This code we have done +1 on reference of an object which is wrong.

- There is an error in merge function in which we have done left++ And right- -.

  Corrected Part of code :

```
public static void mergeSort(int[] array) {
    if (array.length > 1) {
        // split array into two halves
        int[] left = leftHalf(array);
        int[] right = rightHalf(array);

        // recursively sort the two halves
        mergeSort(left);
        mergeSort(right);

        // merge the sorted halves into a sorted whole
        merge(array, left, right);
    }
}
```

6. Matrix Multiplication:

- There is an error in the code in the loop (3rd nested loop) in computing the sum . The correct sum is:sum += first[c][k] * second[k][d];
- Also in this loop k bound is also wrong.k<n is the correct bound.
  Corrected Part of the code:

```
for (c = 0; c < m; c++) {
   for (d = 0; d < q; d++) {
      for (k = 0; k < n; k++) {  // Fix k bound to 'n'
         sum += first[c][k] * second[k][d];  // Fix indexing
      }
      multiply[c][d] = sum;
      sum = 0;  // Reset sum for the next element
   }
}
```

7.Quadratic Probing:

- There is an error in insert function.where we are calculation the probing. It should be like this : i = (i + h * h) % maxSize
- Also increment the h instead of decrement.
- There is an error in remove() function . where we are decreasing the current size by 1 in loop . But I should be outside of the loop only.

  Corrected part of the code:

```
public void insert(String key, String val)
{
    int tmp = hash(key);
    int i = tmp, h = 1;
    do
    {
        if (keys[i] == null)
        {
            keys[i] = key;
            vals[i] = val;
            currentSize++;
            return;
        }
        if (keys[i].equals(key))
        {
            vals[i] = val;
            return;
        }
        i + = (i + h*h) % maxSize;
        h++;
    } while (i != tmp);
}
```

```java
public void remove(String key)
{
    if (!contains(key))
        return;
    /** find position key and delete **/

    int i = hash(key), h = 1;

    while (!key.equals(keys[i]))

        i = (i + h * h++) % maxSize;

    keys[i] = vals[i] = null;
    /** rehash all keys **/
    for (i = (i + h * h++) % maxSize; keys[i] != null; i = (i + h * h++) % maxSize)
    {
        String tmp1 = keys[i], tmp2 = vals[i];
        keys[i] = vals[i] = null;
        insert(tmp1, tmp2);
    }
    currentSize--;
}
```

8. Short the Array

- Here there is an error in the outside loop. The loop is wrong.
- And the swap condition is wrong . Because we want to short the Array in ascending order.so the condition should be like this : if(a[i]>a[j]) then swap.

Corrected Code:

```
int n, temp;
Scanner s = new Scanner(System.in);
System.out.print("Enter no. of elements you want in array:");
n = s.nextInt();
int a[] = new int[n];
System.out.println("Enter all the elements:");
for (int i = 0; i < n; i++)
{
   a[i] = s.nextInt();
}
for (int i = 0; i <n; i++);
{
    for (int j = i + 1; j < n; j++)
    {
        if (a[i] > a[j])
        {
            temp = a[i];
            a[i] = a[j];
            a[j] = temp;
        }
    }
}
System.out.print("Ascending Order:");
for (int i = 0; i < n - 1; i++)
{
    System.out.print(a[i] + ",");
}
System.out.print(a[n - 1]);
```

9.Stack implementation:

- There are errors in push method and pop method.
- In push method when we add elements ,we have to do top++
- And when we pop the elements , we have to do top –.
- There is an error in display method also.

- To fix this issue we need to set breakpoints at push pop and Display methods.

Corrected Code:

```java
public void push(int value){
    if(top==size-1){
        System.out.println("Stack is full, can't push a value");
    }
    else{
        top++;
        stack[top]=value;
    }
}

public void pop(){
    if(!isEmpty())
        top--;
    else{
        System.out.println("Can't pop...stack is empty");
    }
}

public boolean isEmpty(){
    return top==-1;
}

public void display(){

    for(int i=top;i>=0;i--){
        System.out.print(stack[i]+ " ");
    }
    System.out.println();
}
```

10.Tower of Hanoi:

There is an error in the last recursive statement.where we have done some unnecessary increments and decrements.

Corrected Code:

```java
public class MainClass {
    public static void main(String[] args) {
        int nDisks = 3;
        doTowers(nDisks, 'A', 'B', 'C');
    }
    public static void doTowers(int topN, char from,
    char inter, char to) {
        if (topN == 1){
            System.out.println("Disk 1 from "
            + from + " to " + to);
        }else {
            doTowers(topN - 1, from, to, inter);
            System.out.println("Disk "
            + topN + " from " + from + " to " + to);
            doTowers(topN-1, inter, from, to)
        }
    }
}
```

Repo Used:GitHub - shouryaj98/Hotel-Management-Project-Java: It is a Hotel Management tool which can be used to manage activites like storing customer details, booking rooms of four different types, ordering food for particular rooms, unbooking rooms and showing the bill.

Q.1 Code Inspection :

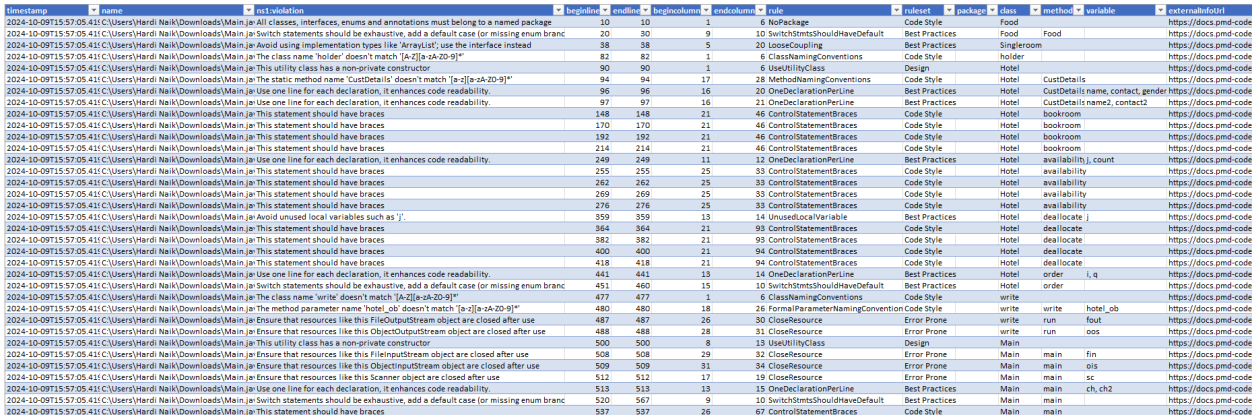
Category B :
-> In class Food the price should be double data type because it can lead to errors.

->In some classes Variables like `name`, `gender` and contact are public by default; encapsulation could be improved by making them private and adding getters/setters.

Category : E
->There should be default case in switch which is missing at some places.

Q.3 Static Analysis Using PMD TOOL

Xls sheet :



timestamp	name	ns1:violation	beginline	endline	begincolumn	endcolumn	rule	ruleset	package	class	method	variable	externalInfoUrl
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	All classes, interfaces, enums and annotations must belong to a named package	10	10	1	6	NoPackage	Code Style		Food			https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Switch statements should be exhaustive, add a default case (or missing enum branc	20	30	9	10	SwitchStmtsShouldHaveDefault	Best Practices	Food	Food			https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Avoid using implementation types like 'ArrayList'; use the interface instead	38	38	5	20	LooseCoupling	Best Practices		Singleroom			https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	The class name 'holder' doesn't match '[A-Z][a-zA-Z0-9]*'	82	82	1	6	ClassNamingConventions	Code Style		holder			https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This utility class has a non-private constructor	90	90	1	6	UseUtilityClass	Design		Hotel			https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	The static method name 'CustDetails' doesn't match '[a-z][a-zA-Z0-9]*'	94	94	17	28	MethodNamingConventions	Code Style		Hotel	CustDetails		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Use one line for each declaration, it enhances code readability.	96	96	16	20	OneDeclarationPerLine	Best Practices		Hotel	CustDetails	name, contact, gender	https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Use one line for each declaration, it enhances code readability.	97	97	16	21	OneDeclarationPerLine	Best Practices		Hotel	CustDetails	name2, contact2	https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This statement should have braces	148	148	21	46	ControlStatementBraces	Code Style		Hotel	bookroom		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This statement should have braces	170	170	21	46	ControlStatementBraces	Code Style		Hotel	bookroom		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This statement should have braces	192	192	21	46	ControlStatementBraces	Code Style		Hotel	bookroom		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This statement should have braces	214	214	21	46	ControlStatementBraces	Code Style		Hotel	bookroom		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Use one line for each declaration, it enhances code readability.	249	249	11	12	OneDeclarationPerLine	Best Practices		Hotel	availability	j, count	https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This statement should have braces	255	255	25	33	ControlStatementBraces	Code Style		Hotel	availability		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This statement should have braces	262	262	25	33	ControlStatementBraces	Code Style		Hotel	availability		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This statement should have braces	269	269	25	33	ControlStatementBraces	Code Style		Hotel	availability		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This statement should have braces	276	276	25	33	ControlStatementBraces	Code Style		Hotel	availability		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Avoid unused local variables such as 'j'.	359	359	13	14	UnusedLocalVariable	Best Practices		Hotel	deallocate	j	https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This statement should have braces	364	364	21	93	ControlStatementBraces	Code Style		Hotel	deallocate		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This statement should have braces	382	382	21	93	ControlStatementBraces	Code Style		Hotel	deallocate		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This statement should have braces	400	400	21	94	ControlStatementBraces	Code Style		Hotel	deallocate		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This statement should have braces	418	418	21	94	ControlStatementBraces	Code Style		Hotel	deallocate		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Use one line for each declaration, it enhances code readability.	441	441	13	14	OneDeclarationPerLine	Best Practices		Hotel	order	i, q	https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Switch statements should be exhaustive, add a default case (or missing enum branc	451	460	15	10	SwitchStmtsShouldHaveDefault	Best Practices		Hotel	order		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	The class name 'write' doesn't match '[A-Z][a-zA-Z0-9]*'	477	477	1	6	ClassNamingConventions	Code Style		write			https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	The method parameter name 'hotel_ob' doesn't match '[a-z][a-zA-Z0-9]*'	480	480	18	26	FormalParameterNamingConvention	Code Style		write	write	hotel_ob	https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Ensure that resources like this FileOutputStream object are closed after use	487	487	26	30	CloseResource	Error Prone		write	run	fout	https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Ensure that resources like this ObjectOutputStream object are closed after use	488	488	28	31	CloseResource	Error Prone		write	run	oos	https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This utility class has a non-private constructor	500	500	8	13	UseUtilityClass	Design		Main			https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Ensure that resources like this FileInputStream object are closed after use	508	508	29	32	CloseResource	Error Prone		Main	main	fin	https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Ensure that resources like this ObjectInputStream object are closed after use	509	509	31	34	CloseResource	Error Prone		Main	main	ois	https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Ensure that resources like this Scanner object are closed after use	512	512	17	19	CloseResource	Error Prone		Main	main	sc	https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Use one line for each declaration, it enhances code readability.	513	513	13	15	OneDeclarationPerLine	Best Practices		Main	main	ch, ch2	https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	Switch statements should be exhaustive, add a default case (or missing enum branc	520	567	9	10	SwitchStmtsShouldHaveDefault	Best Practices		Main	main		https://docs.pmd-code
2024-10-09T15:57:05.41	C:\Users\Hardi Naik\Downloads\Main.ja	This statement should have braces	537	537	26	67	ControlStatementBraces	Code Style		Main	main		https://docs.pmd-code