اونيۏرسيتي تيکنولوڬي مارا

UNIVERSITI TEKNOLOGI MARA

CSC248
FUNDAMENTALS OF DATA STRUCTURE

LAB ASSIGNMENT 4

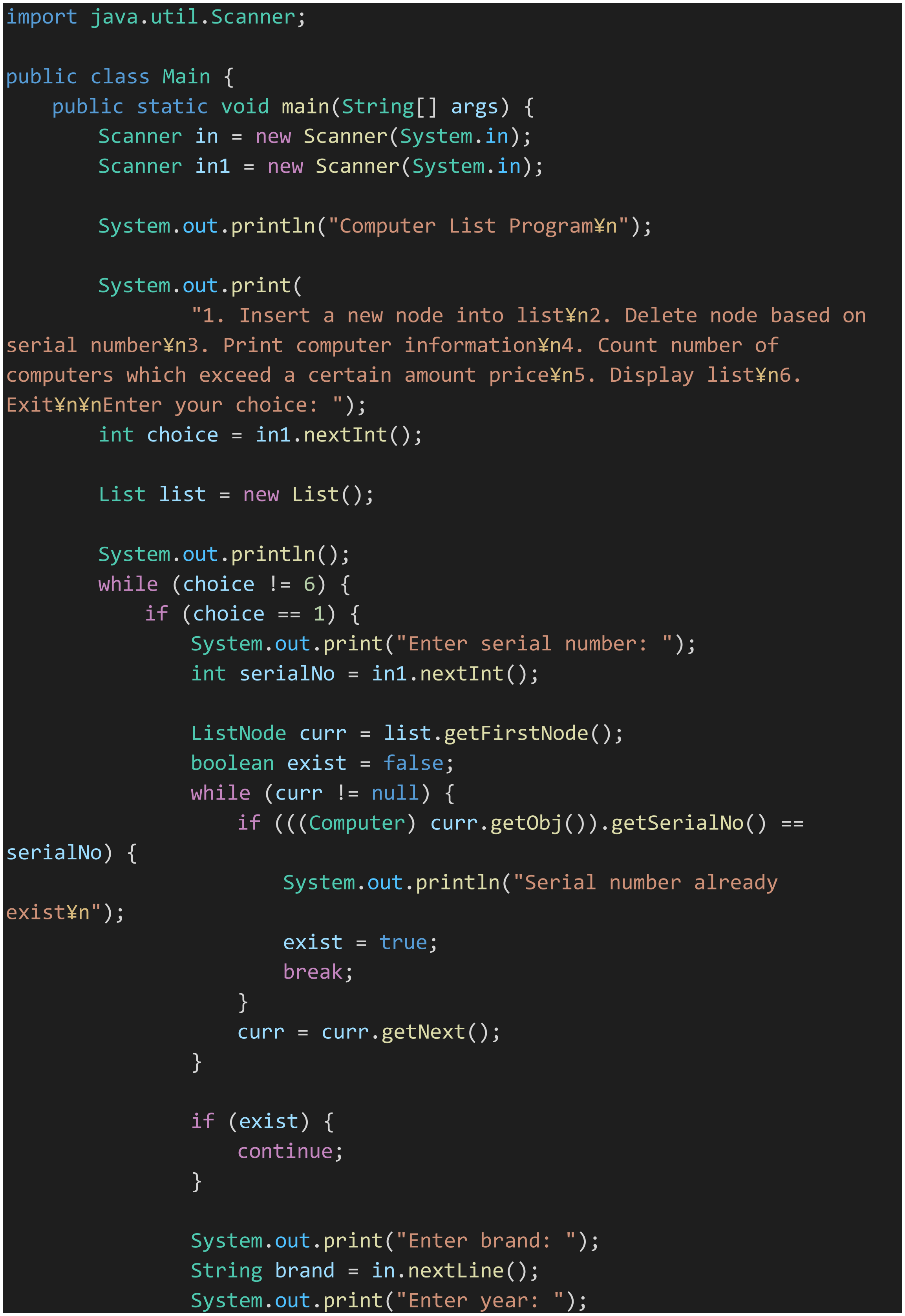
NAME : MUHAMMAD REDZA BIN MAHAYADIN

STUDENT ID : 2022676696

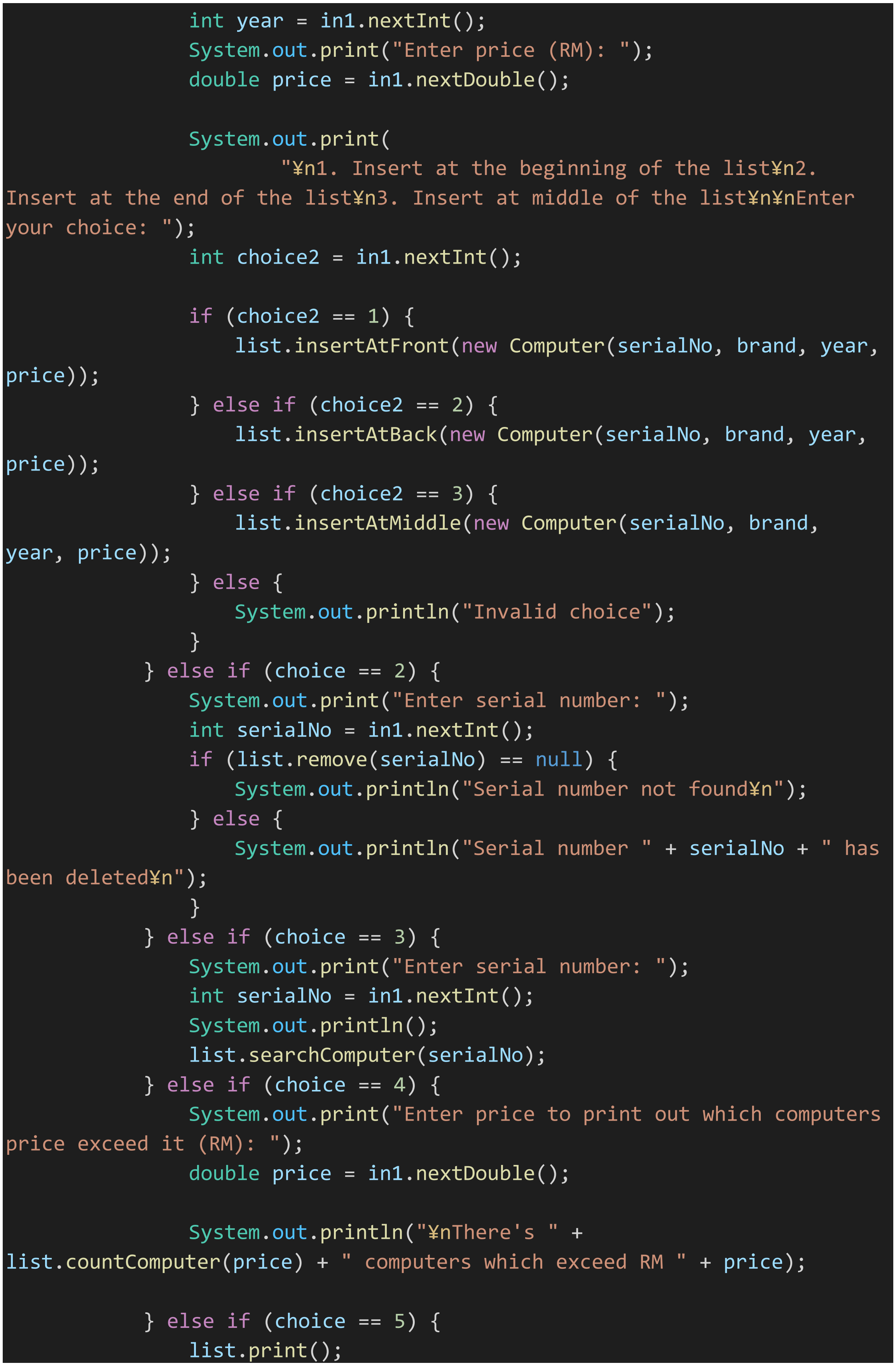
GROUP : RCDCS1103B

LECTURER : SIR MOHD NIZAM BIN OSMAN

Main class



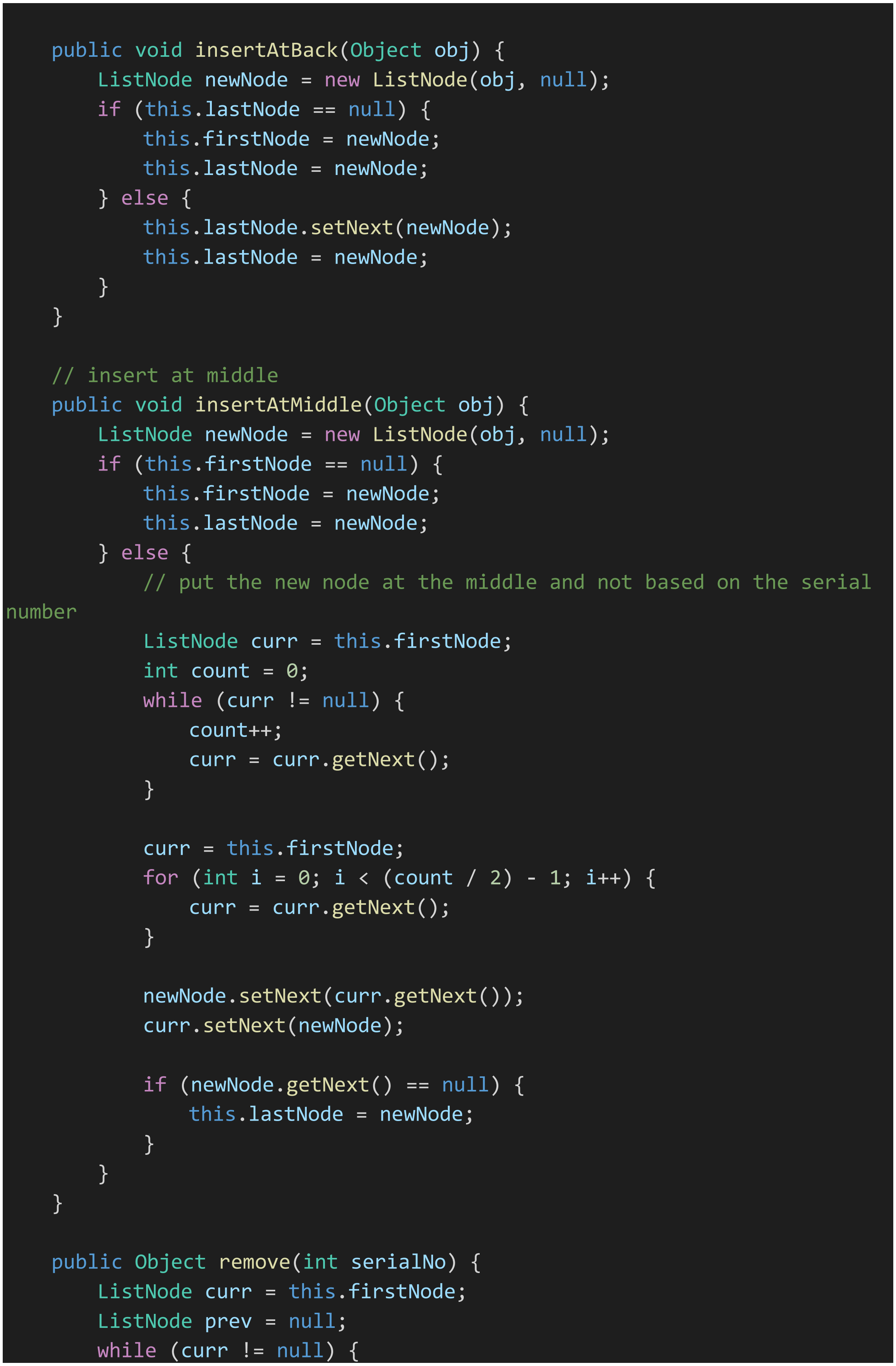
```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        Scanner in1 = new Scanner(System.in);

        System.out.println("Computer List Program¥n");

        System.out.print(
                "1. Insert a new node into list¥n2. Delete node based on
serial number¥n3. Print computer information¥n4. Count number of
computers which exceed a certain amount price¥n5. Display list¥n6.
Exit¥n¥nEnter your choice: ");
        int choice = in1.nextInt();

        List list = new List();

        System.out.println();
        while (choice != 6) {
            if (choice == 1) {
                System.out.print("Enter serial number: ");
                int serialNo = in1.nextInt();

                ListNode curr = list.getFirstNode();
                boolean exist = false;
                while (curr != null) {
                    if (((Computer) curr.getObj()).getSerialNo() ==
serialNo) {
                        System.out.println("Serial number already
exist¥n");
                        exist = true;
                        break;
                    }
                    curr = curr.getNext();
                }

                if (exist) {
                    continue;
                }

                System.out.print("Enter brand: ");
                String brand = in.nextLine();
                System.out.print("Enter year: ");
```



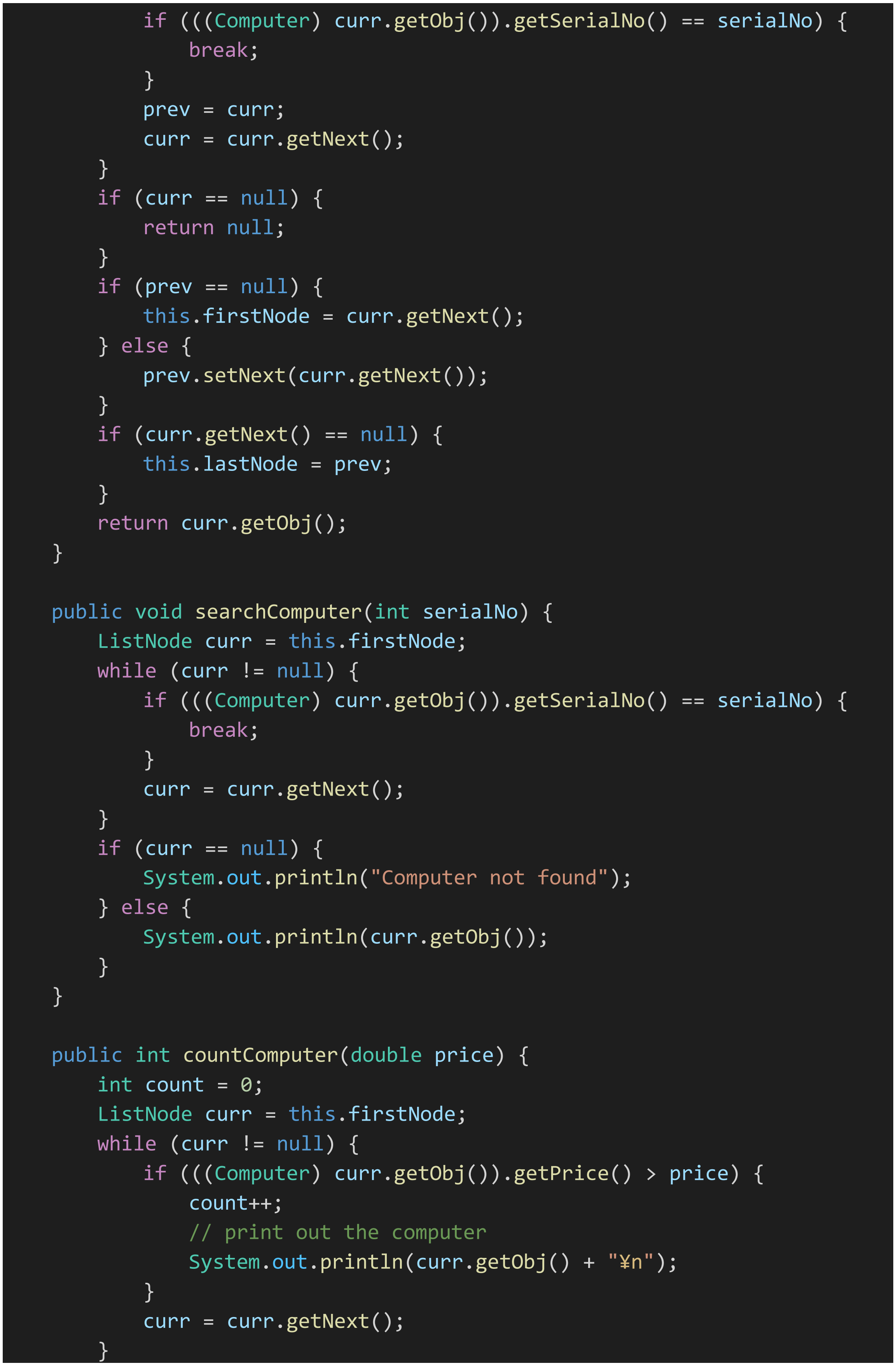
```
                int year = in1.nextInt();
                System.out.print("Enter price (RM): ");
                double price = in1.nextDouble();

                System.out.print(
                        "¥n1. Insert at the beginning of the list¥n2. 
Insert at the end of the list¥n3. Insert at middle of the list¥n¥nEnter 
your choice: ");
                int choice2 = in1.nextInt();

                if (choice2 == 1) {
                    list.insertAtFront(new Computer(serialNo, brand, year, 
price));
                } else if (choice2 == 2) {
                    list.insertAtBack(new Computer(serialNo, brand, year, 
price));
                } else if (choice2 == 3) {
                    list.insertAtMiddle(new Computer(serialNo, brand, 
year, price));
                } else {
                    System.out.println("Invalid choice");
                }
            } else if (choice == 2) {
                System.out.print("Enter serial number: ");
                int serialNo = in1.nextInt();
                if (list.remove(serialNo) == null) {
                    System.out.println("Serial number not found¥n");
                } else {
                    System.out.println("Serial number " + serialNo + " has 
been deleted¥n");
                }
            } else if (choice == 3) {
                System.out.print("Enter serial number: ");
                int serialNo = in1.nextInt();
                System.out.println();
                list.searchComputer(serialNo);
            } else if (choice == 4) {
                System.out.print("Enter price to print out which computers 
price exceed it (RM): ");
                double price = in1.nextDouble();

                System.out.println("¥nThere's " + 
list.countComputer(price) + " computers which exceed RM " + price);

            } else if (choice == 5) {
                list.print();
            } else {
                System.out.println("Invalid choice");
            }
            System.out.print(
                    "1. Insert a new node into list¥n2. Delete node based
on serial number¥n3. Print computer information¥n4. Count number of
computers which exceed a certain amount price¥n5. Display list¥n6.
Exit¥n¥nEnter your choice: ");
            choice = in1.nextInt();
            System.out.println();
        }

        in.close();
        in1.close();

        System.out.println("Program terminated.");
    }
}
```

List Class

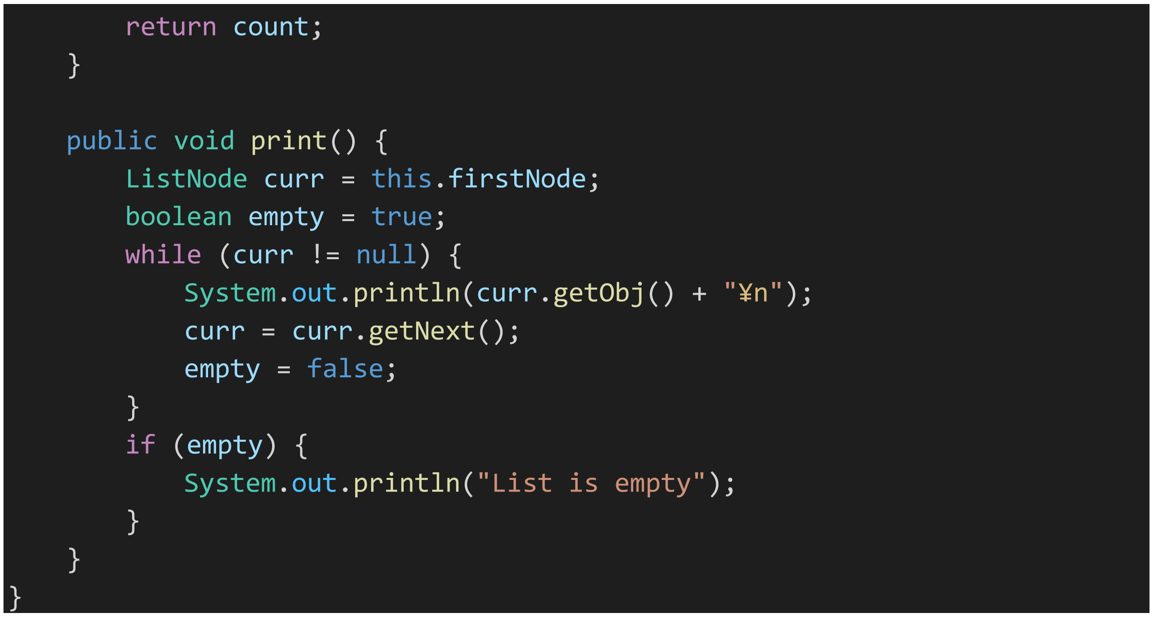
```java
public class List {
    private ListNode firstNode;
    private ListNode lastNode;
    private ListNode currNode;

    public List() {
        this.firstNode = null;
        this.lastNode = null;
        this.currNode = null;
    }

    public ListNode getFirstNode() {
        return this.firstNode;
    }

    public void setFirstNode(ListNode firstNode) {
        this.firstNode = firstNode;
    }

    public ListNode getLastNode() {
        return this.lastNode;
    }

    public void setLastNode(ListNode lastNode) {
        this.lastNode = lastNode;
    }

    public ListNode getCurrNode() {
        return this.currNode;
    }

    public void setCurrNode(ListNode currNode) {
        this.currNode = currNode;
    }

    public void insertAtFront(Object obj) {
        ListNode newNode = new ListNode(obj, this.firstNode);
        this.firstNode = newNode;
        if (this.lastNode == null) {
            this.lastNode = newNode;
        } else {
            this.lastNode.setNext(newNode);
        }
    }
```



```
    public void insertAtBack(Object obj) {
        ListNode newNode = new ListNode(obj, null);
        if (this.lastNode == null) {
            this.firstNode = newNode;
            this.lastNode = newNode;
        } else {
            this.lastNode.setNext(newNode);
            this.lastNode = newNode;
        }
    }

    // insert at middle
    public void insertAtMiddle(Object obj) {
        ListNode newNode = new ListNode(obj, null);
        if (this.firstNode == null) {
            this.firstNode = newNode;
            this.lastNode = newNode;
        } else {
            // put the new node at the middle and not based on the serial
number
            ListNode curr = this.firstNode;
            int count = 0;
            while (curr != null) {
                count++;
                curr = curr.getNext();
            }

            curr = this.firstNode;
            for (int i = 0; i < (count / 2) - 1; i++) {
                curr = curr.getNext();
            }

            newNode.setNext(curr.getNext());
            curr.setNext(newNode);

            if (newNode.getNext() == null) {
                this.lastNode = newNode;
            }
        }
    }

    public Object remove(int serialNo) {
        ListNode curr = this.firstNode;
        ListNode prev = null;
        while (curr != null) {
```



```java
            if (((Computer) curr.getObj()).getSerialNo() == serialNo) {
                break;
            }
            prev = curr;
            curr = curr.getNext();
        }
        if (curr == null) {
            return null;
        }
        if (prev == null) {
            this.firstNode = curr.getNext();
        } else {
            prev.setNext(curr.getNext());
        }
        if (curr.getNext() == null) {
            this.lastNode = prev;
        }
        return curr.getObj();
    }

    public void searchComputer(int serialNo) {
        ListNode curr = this.firstNode;
        while (curr != null) {
            if (((Computer) curr.getObj()).getSerialNo() == serialNo) {
                break;
            }
            curr = curr.getNext();
        }
        if (curr == null) {
            System.out.println("Computer not found");
        } else {
            System.out.println(curr.getObj());
        }
    }

    public int countComputer(double price) {
        int count = 0;
        ListNode curr = this.firstNode;
        while (curr != null) {
            if (((Computer) curr.getObj()).getPrice() > price) {
                count++;
                // print out the computer
                System.out.println(curr.getObj() + "¥n");
            }
            curr = curr.getNext();
        }
```



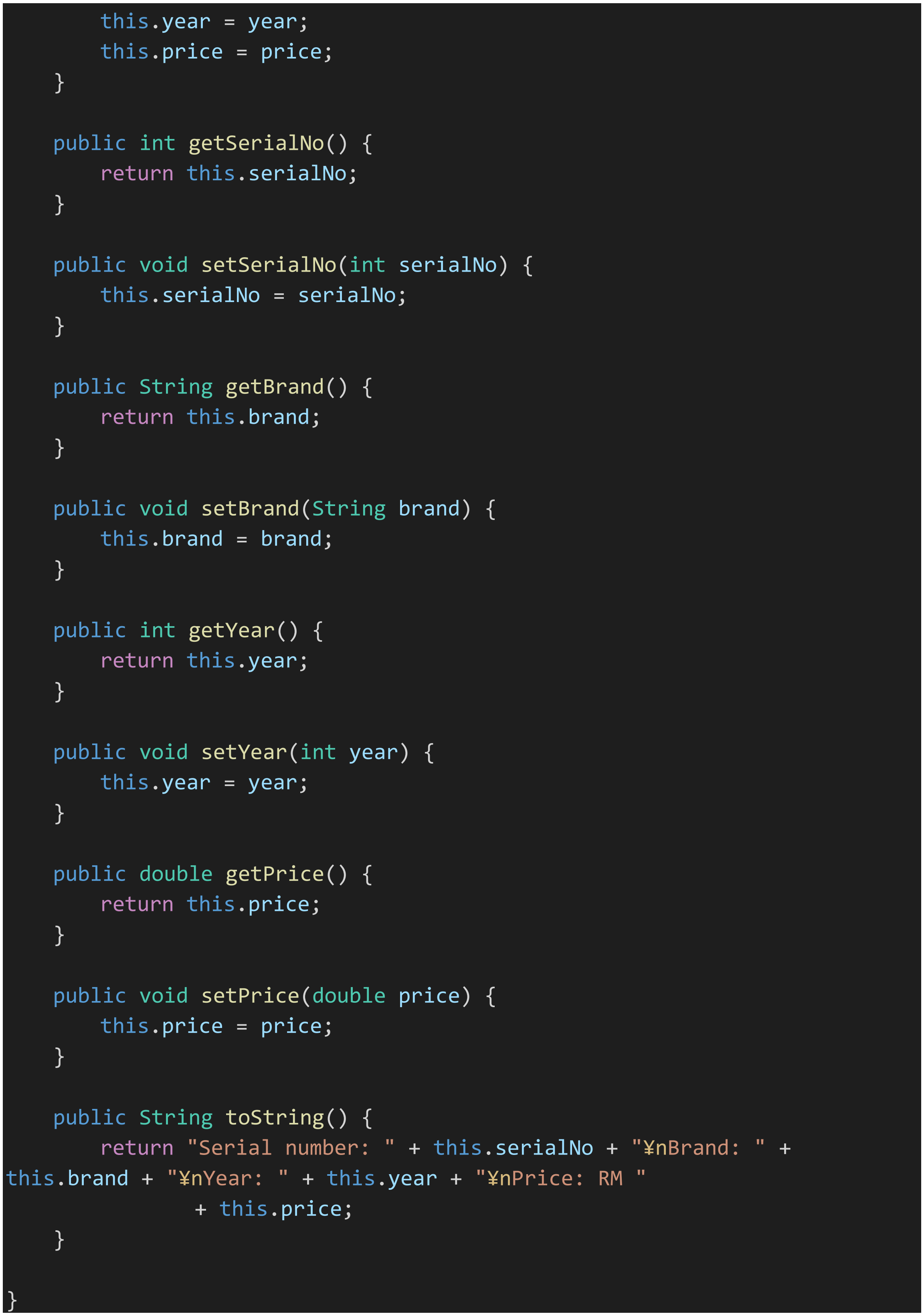
```java
        return count;
    }

    public void print() {
        ListNode curr = this.firstNode;
        boolean empty = true;
        while (curr != null) {
            System.out.println(curr.getObj() + "¥n");
            curr = curr.getNext();
            empty = false;
        }
        if (empty) {
            System.out.println("List is empty");
        }
    }
}
```

ListNode class

```
public class ListNode {
    private Object obj;
    private ListNode next;

    public ListNode(Object obj, ListNode next) {
        this.obj = obj;
        this.next = next;
    }

    public Object getObj() {
        return this.obj;
    }

    public void setObj(Object obj) {
        this.obj = obj;
    }

    public ListNode getNext() {
        return this.next;
    }

    public void setNext(ListNode next) {
        this.next = next;
    }

    public String toString() {
        return this.obj.toString();
    }
}
```

Computer class

```
public class Computer {
    private int serialNo;
    private String brand;
    private int year;
    private double price;

    public Computer(int serialNo, String brand, int year, double price) {
        this.serialNo = serialNo;
        this.brand = brand;
```



```
        this.year = year;
        this.price = price;
    }

    public int getSerialNo() {
        return this.serialNo;
    }

    public void setSerialNo(int serialNo) {
        this.serialNo = serialNo;
    }

    public String getBrand() {
        return this.brand;
    }

    public void setBrand(String brand) {
        this.brand = brand;
    }

    public int getYear() {
        return this.year;
    }

    public void setYear(int year) {
        this.year = year;
    }

    public double getPrice() {
        return this.price;
    }

    public void setPrice(double price) {
        this.price = price;
    }

    public String toString() {
        return "Serial number: " + this.serialNo + "¥nBrand: " +
this.brand + "¥nYear: " + this.year + "¥nPrice: RM "
                + this.price;
    }

}
```