Question 1:

In a scheduling application, users often need to view upcoming events. The application utilizes an array to store event IDs, where each ID corresponds to a unique event. Users can input the number of events, the event IDs, and specify the number of positions to right rotate the array for efficient event viewing.

Input Format:

- The user is prompted to enter the number of elements in the array.
- Then, the user is prompted to enter each element of the array one by one.
- Finally, the user is prompted to enter the number of positions to rotate the array to the right.

Output Format:

- The program outputs the array after performing the right rotation.
- First, it displays "Array after right rotation: " followed by a space-separated list of elements in the array after the rotation.

Title for Question 1: Implementing and Demonstrating Right Rotation of an Array

Solution:

```
#include <iostream>
using namespace std;

void rightRotateByOne(int arr[], int n) {
    int last = arr[n - 1];
    for (int i = n - 1; i > 0; i--) {
        arr[i] = arr[i - 1];
    }
    arr[0] = last;
}

// Function to right rotate an array by d positions
void rightRotate(int arr[], int d, int n) {
    d = d % n; // To handle d > n
    for (int i = 0; i < d; i++)
        rightRotateByOne(arr, n);
}

// Utility function to print the elements of an array
void printArray(int arr[], int n) {
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;
}

int main() {
    int n, d;
```

```
    //cout << "Enter the number of elements in the array: ";
    cin >> n;

    int arr[n]; // Create an array of size n

    //cout << "Enter " << n << " elements of the array:\n";
    for(int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    //cout << "Enter the number of positions to rotate the array to the r
    cin >> d;

    rightRotate(arr, d, n);

    cout << "Array after right rotation: ";
    printArray(arr, n);

    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	5 1 2 3 4 5 2	Array after right rotation: 4 5 1 2 3
2	6 10 20 30 40 50 60 3	Array after right rotation: 40 50 60 10 20 30
3	4 7 8 9 10 1	Array after right rotation: 10 7 8 9
4	3 -1 -2 -3 5	Array after right rotation: -2 -3 -1
5	8 100 200 300 400 500 600 700 800 4	Array after right rotation: 500 600 700 800 100 200 300 400
6	7 -10 -20 -30 -40 -50 -60 -70 2	Array after right rotation: -60 -70 -10 -20 -30 -40 -50

White List:

Black List:

Question 2:

Suppose you're developing a program to analyze and process user-inputted strings, particularly focusing on extracting and displaying substrings that end with vowels (a, e, i, o, u).

Input Format:

- The input consists of a single line containing a string s.

Output Format:

- For each vowel (a, e, i, o, u) found in the input string, the program prints a new line containing the substring of s starting from that vowel till the end of the string.

Title for Question 2: Reverse Vowel Segmentation and Printing in a String

Solution:

```
#include<iostream>
using namespace std;
int main()
{
    string s;
    cin>>s;
    int a=s.length();
    for(int i=a-1;i>=0;i--)
    {
        if(s[i]=='a' ||s[i]=='e' ||s[i]=='i' ||s[i]=='o' ||s[i]=='u')
        {
            for(int j=i;s[j]!='\0';j++)
            cout<<s[j];
            s[i]='\0';
            cout<<"\n";
        }

    }
}
```

TestCases:

S.No	Inputs	Outputs
1	programming	ing amm ogr
2	algorithm	ithm or alg
3	banana	a an an
4	hello	o ell
5	Everything	ing eryth
6	testing	ing est

White List:

Black List:

Question 3:

In a university record system, a Doubly Linked List is used to manage student records. Each node represents a student with details like name, ID, and GPA. Upon request for student withdrawal, the system deletes the record at a specified position, adjusting pointers accordingly, ensuring data integrity in the linked list.

Input Format:

- The user inputs an integer n representing the number of nodes in the doubly linked list.
- Then, the user inputs n integers representing the elements of the doubly linked list.
- Finally, the user inputs an integer position representing the position of the node to be deleted.
-

Output Format:

- The program displays the doubly linked list after deletion of the node at the specified position.

Title for Question 3: Implementing a Doubly Linked List : Deletion by Position

Solution:

```
#include <iostream>
using namespace std;

class Node {
public:
    int data;
    Node* prev;
    Node* next;

    Node(int data) : data(data), prev(nullptr), next(nullptr) {}
};

class DoublyLinkedList {
public:
    Node* head;

    DoublyLinkedList() : head(nullptr) {}

    // Function to add node at the end of the list
    void append(int data) {
        Node* newNode = new Node(data);
        if (head == nullptr) {
            head = newNode;
            return;
        }
        Node* temp = head;
        while (temp->next != nullptr) {
            temp = temp->next;
        }
        temp->next = newNode;
        newNode->prev = temp;
    }

    // Function to delete node at a given position
    void deleteNodeAtPosition(int position) {
        if (head == nullptr) {
            cout << "List is empty." << endl;
            return;
        }
        Node* temp = head;
        if (position == 1) {
```

```
            head = temp->next;
            if (head != nullptr) {
                head->prev = nullptr;
            }
            delete temp;
            return;
        }
        for (int i = 1; temp != nullptr && i < position; i++) {
            temp = temp->next;
        }
        if (temp == nullptr) {
            cout << "Position exceeds the size of the doubly linked list.
            return;
        }
        if (temp->next != nullptr) {
            temp->next->prev = temp->prev;
        }
        if (temp->prev != nullptr) {
            temp->prev->next = temp->next;
        }
        delete temp;
    }

    // Function to display the list
    void display() {
        Node* temp = head;
        while (temp != nullptr) {
            cout << temp->data << " ";
            temp = temp->next;
        }
        cout << endl;
    }
};

int main() {
    DoublyLinkedList dll;
    int n, data, position;
    //cout << "Enter the number of nodes: ";
    cin >> n;
    //cout << "Enter " << n << " elements:" << endl;
    for (int i = 0; i < n; i++) {
        cin >> data;
        dll.append(data);
    }
    //cout << "Enter the position of the node to delete: ";
    cin >> position;
    dll.deleteNodeAtPosition(position);
    cout << "Doubly Linked List after deletion:" << endl;
    dll.display();
    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	5 1 2 3 4 5 3	Doubly Linked List after deletion: 1 2 4 5
2	4 10 20 30 40 1	Doubly Linked List after deletion: 20 30 40

S.No	Inputs	Outputs
3	6 5 10 15 20 25 30 6	Doubly Linked List after deletion: 5 10 15 20 25
4	3 100 200 300 2	Doubly Linked List after deletion: 100 300
5	0 1	List is empty. Doubly Linked List after deletion:
6	6 1 3 5 7 9 11 5	Doubly Linked List after deletion: 1 3 5 7 11

White List:

Black List:

Question 4:

In a computer science workshop, students are learning about binary numbers. A program is developed to convert decimal numbers to binary representation using a stack. As students input decimal numbers, the program efficiently converts them into binary format, aiding in their understanding of binary arithmetic and computer memory concepts.

Input Format:

- The user is prompted to enter a decimal number.
- The decimal number is read from the standard input.

Output Format:

- The decimal number entered by the user is displayed.
- The binary representation of the decimal number is shown on the same line.

Title for Question 4: Decimal to Binary Converter

Solution:

```
#include <iostream>
using namespace std;

class Stack {
private:
    int top;
    int MaxSize;
    int* stackarray;

public:
    Stack() : top(-1), MaxSize(100) {
        stackarray = new int[MaxSize];
    }

    ~Stack() {
        delete[] stackarray;
    }
```

```
    void push(int a) {
        if(top < MaxSize - 1) {
            stackarray[++top] = a;
        } else {
            cout << "Stack overflow" << endl;
        }
    }

    void pop() {
        while(top != -1) {
            cout << stackarray[top--];
        }
    }
};

int main() {
    Stack st;
    int num;
    //cout << "Enter a decimal number: ";
    cin >> num;
    cout << "Decimal Number: " << num << endl;
    cout << "Binary Representation: ";

    if(num == 0) st.push(0); // Handle the case for 0 explicitly

    while(num != 0) {
        st.push(num % 2);
        num /= 2;
    }

    st.pop();
    cout << endl;

    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	10	Decimal Number: 10 Binary Representation: 1010
2	25	Decimal Number: 25 Binary Representation: 11001
3	0	Decimal Number: 0 Binary Representation: 0
4	7	Decimal Number: 7 Binary Representation: 111
5	50	Decimal Number: 50 Binary Representation: 110010
6	128	Decimal Number: 128 Binary Representation: 10000000

White List:

Black List:

Question 5:

In a retail checkout system, postfix expressions can be used to calculate total prices. Each item's price and quantity are scanned, and operators like '+' for addition or '*' for multiplication can be

input to apply discounts or calculate the final bill. The system processes the postfix expression to determine the total amount due.

Input Format:

- User need to enter the postfix expression containing integers and arithmetic operators separated by spaces.

Output Format:

- The program will output the result of evaluating the postfix expression. If any errors occur during evaluation (such as division by zero or invalid operations), appropriate error messages will be displayed.

Title for Question 5: Evaluating Postfix Expressions Using a Stack

Solution:

```
#include <iostream>
#include <string>
#include <cctype> // for isdigit function

using namespace std;

class Stack {
private:
    int top;
    int maxSize;
    int* stackArray;

public:
    Stack(int size) : maxSize(size), top(-1) {
        stackArray = new int[maxSize];
    }

    ~Stack() {
        delete[] stackArray;
    }

    bool isFull() const {
        return top == maxSize - 1;
    }

    bool isEmpty() const {
        return top == -1;
    }

    void push(int item) {
        if (!isFull()) {
            stackArray[++top] = item;
        } else {
            //cout << "Stack is full." << endl;
```

```
        }
    }

    int pop() {
        if (!isEmpty()) {
            return stackArray[top--];
        } else {
            //cout << "Stack is empty." << endl;
            return -1; // Returning -1 to indicate the stack is empty
        }
    }

    void print() const {
        if (!isEmpty()) {
            cout << "Final Answer: " << stackArray[top] << endl;
        } else {
            cout << "Stack is empty." << endl;
        }
    }
};

int main() {
    string input;
    //cout << "Enter postfix expression: ";
    getline(cin, input);
    Stack st(input.length());

    for (char c : input) {
        if (isdigit(c)) {
            st.push(c - '0'); // Convert character to integer
        } else {
            int val2 = st.pop(); // It's important to pop val2 before val
            int val1 = st.pop();
            switch (c) {
                case '+':
                    st.push(val1 + val2);
                    break;
                case '-':
                    st.push(val1 - val2);
                    break;
                case '*':
                    st.push(val1 * val2);
                    break;
                case '/':
                    if (val2 != 0) {
                        st.push(val1 / val2);
                    } else {
                        cout << "Cannot divide by zero" << endl;
                    }
                    break;
                default:
                    cout << "Invalid operation" << endl;
            }
        }
    }

    st.print();
    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	54+2*	Final Answer: 18
2	832*+	Final Answer: 14
3	12+34*-	Final Answer: -9
4	35*61+/	Final Answer: 2
5	64+2/3*	Final Answer: 15
6	92+58-*	Final Answer: -33

White List:

Black List:

Question 6:

Imagine you are developing a software tool for cryptography or security applications. One common requirement in such applications is the generation of prime numbers within a specified range. Prime numbers play a crucial role in cryptography algorithms, such as RSA, where large prime numbers are used for encryption and decryption processes.

Input Format:

- The user is prompted to enter the lower bound of the range of numbers.
- The user then enters an integer representing the lower bound.
- Next, the user is prompted to enter the upper bound of the range of numbers.
- The user then enters an integer representing the upper bound.

Output Format:

- The program outputs a message asking the user to enter the lower bound of the range.
- The program waits for the user to input the lower bound.
- The program outputs a message asking the user to enter the upper bound of the range.
- The program waits for the user to input the upper bound.
- The program then outputs the prime numbers within the specified range, each on a separate line.

Title for Question 6: Identifying Prime Numbers within a Specified Range

Solution:

```
#include <iostream>

// Function to check if a number is prime
bool isPrime(int num) {
    if (num <= 1) {
        return false; // 0 and 1 are not prime numbers
    }
```

```
    for (int i = 2; i * i <= num; i++) {
        if (num % i == 0) {
            return false; // Found a divisor, so not prime
        }
    }
    return true; // No divisors were found, so it's prime
}

int main() {
    int lower, upper;
    std::cin >> lower; // Reading lower bound of the range
    std::cin >> upper; // Reading upper bound of the range

    std::cout << "Prime numbers between " << lower << " and " << upper <<
    for (int num = lower; num <= upper; num++) {
        if (isPrime(num)) {
            std::cout << num << std::endl; // Printing the prime number
        }
    }

    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	1 10	Prime numbers between 1 and 10 are: 2 3 5 7
2	20 30	Prime numbers between 20 and 30 are: 23 29
3	50 60	Prime numbers between 50 and 60 are: 53 59
4	100 110	Prime numbers between 100 and 110 are: 101 103 107 109
5	0 20	Prime numbers between 0 and 20 are: 2 3 5 7 11 13 17 19
6	30 40	Prime numbers between 30 and 40 are: 31 37

White List:

Black List:
