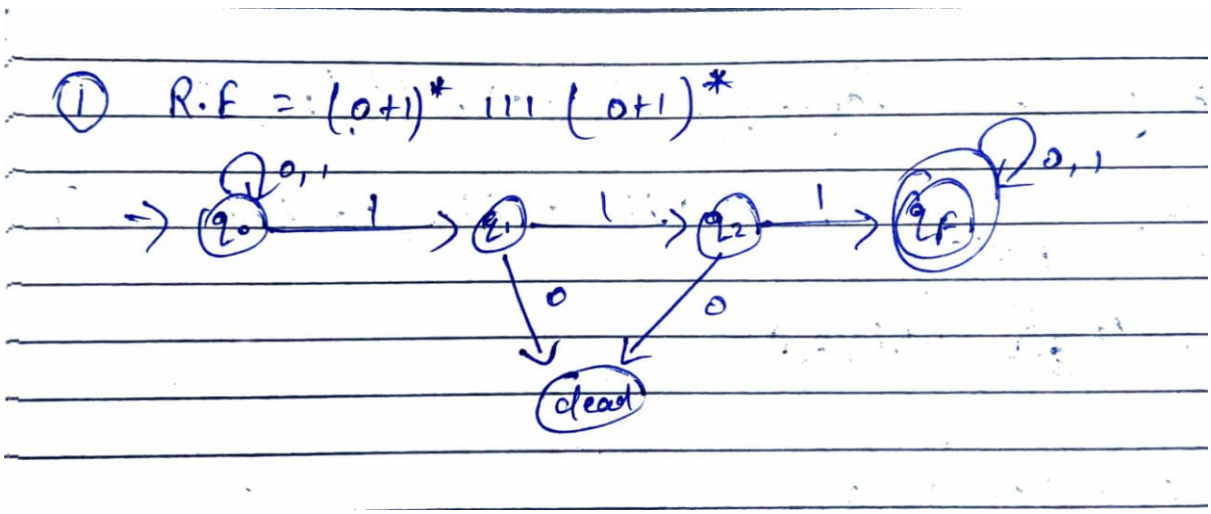
Q1 — FA for 3 consecutive 1's (111)

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

void State0(string w, int i);
void State1(string w, int i);
void State2(string w, int i);
void State3(string w, int i);

int main() {
    string w;
    cout << "Enter a string: ";
    cin >> w;
    State0(w, 0);
    return 0;
}

void State0(string w, int i) {
    if (i == w.length()) {
        cout << "String is rejected\n";
        return;
    }
    if (w[i] == '1')
        State1(w, i + 1);
    else
        State0(w, i + 1);
}

void State1(string w, int i) {
    if (i == w.length()) {
        cout << "String is rejected\n";
        return;
    }
    if (w[i] == '1')
        State2(w, i + 1);
    else
        State0(w, i + 1);
}

void State2(string w, int i) {
    if (i == w.length()) {
        cout << "String is rejected\n";
        return;
```

```
    }
    if (w[i] == '1')
        State3(w, i + 1);
    else
        State0(w, i + 1);
}

void State3(string w, int i) {
    cout << "String is accepted\n";
}
```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                                   Code

-o 1st } ; if ($?) { .\1st }
Enter a string: 0111
State 0
State 0
State 1
State 2
State 3
State 3
String is accepted
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 1st.cpp -o 1st } ; if ($?) { .\1st }
Enter a string: 110
State 0
State 1
State 2
State 0
String is rejected
PS E:\sem_V\toc_prac>
```



Q2 — FA for exactly two or exactly three 1’s

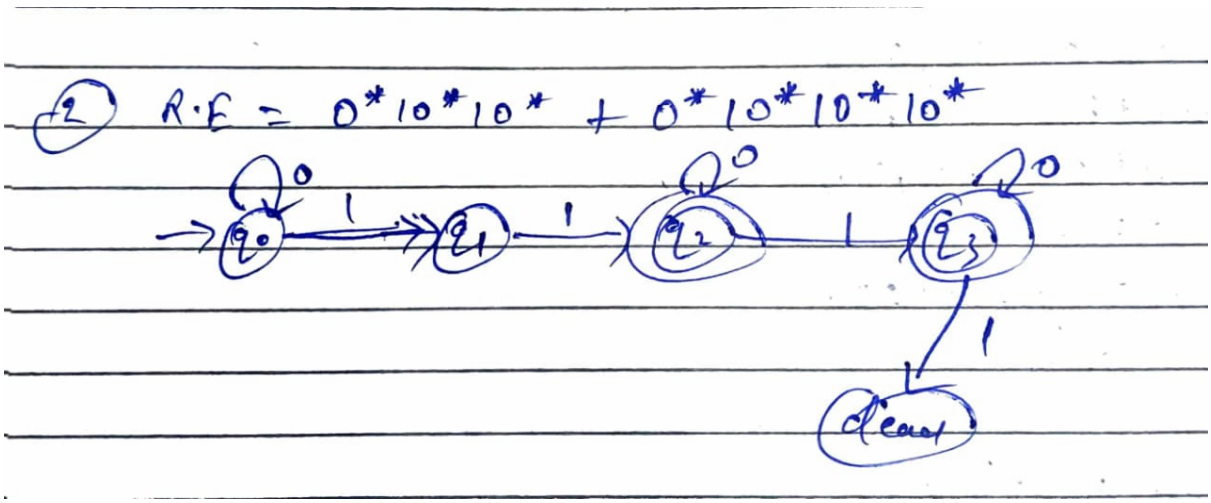
```
#include <iostream>
#include <string>
using namespace std;
```

```
void State0(string w, int i);
void State1(string w, int i);
void State2(string w, int i);
void State3(string w, int i);
void State4(string w, int i);

int main() {
    string w;
    cout << "Enter a binary string: ";
    cin >> w;
    State0(w, 0);
    return 0;
}

void State0(string w, int i) {
    if (i == w.length()) {
        cout << "String is rejected\n";
        return;
    }
    if (w[i] == '1') State1(w, i + 1);
    else State0(w, i + 1);
}

void State1(string w, int i) {
    if (i == w.length()) {
        cout << "String is rejected\n";
        return;
    }
    if (w[i] == '1') State2(w, i + 1);
    else State1(w, i + 1);
}

void State2(string w, int i) {
    if (i == w.length()) {
        cout << "String is accepted\n";
        return;
    }
    if (w[i] == '1') State3(w, i + 1);
    else State2(w, i + 1);
}

void State3(string w, int i) {
    if (i == w.length()) {
        cout << "String is accepted\n";
        return;
    }
    if (w[i] == '1') State4(w, i + 1);
```

```
    else State3(w, i + 1);
}

void State4(string w, int i) {
    if (i == w.length()) {
        cout << "String is rejected\n";
        return;
    }
    State4(w, i + 1);
}
```

Output:

```
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 2nd.cpp -o 2nd } ; if ($?) { .\2nd }
Enter a binary string: 111
String is accepted
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 2nd.cpp -o 2nd } ; if ($?) { .\2nd }
Enter a binary string: 011
String is accepted
PS E:\sem_V\toc_prac> 1
1
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 2nd.cpp -o 2nd } ; if ($?) { .\2nd }
Enter a binary string: 1
String is rejected
PS E:\sem_V\toc_prac>
```



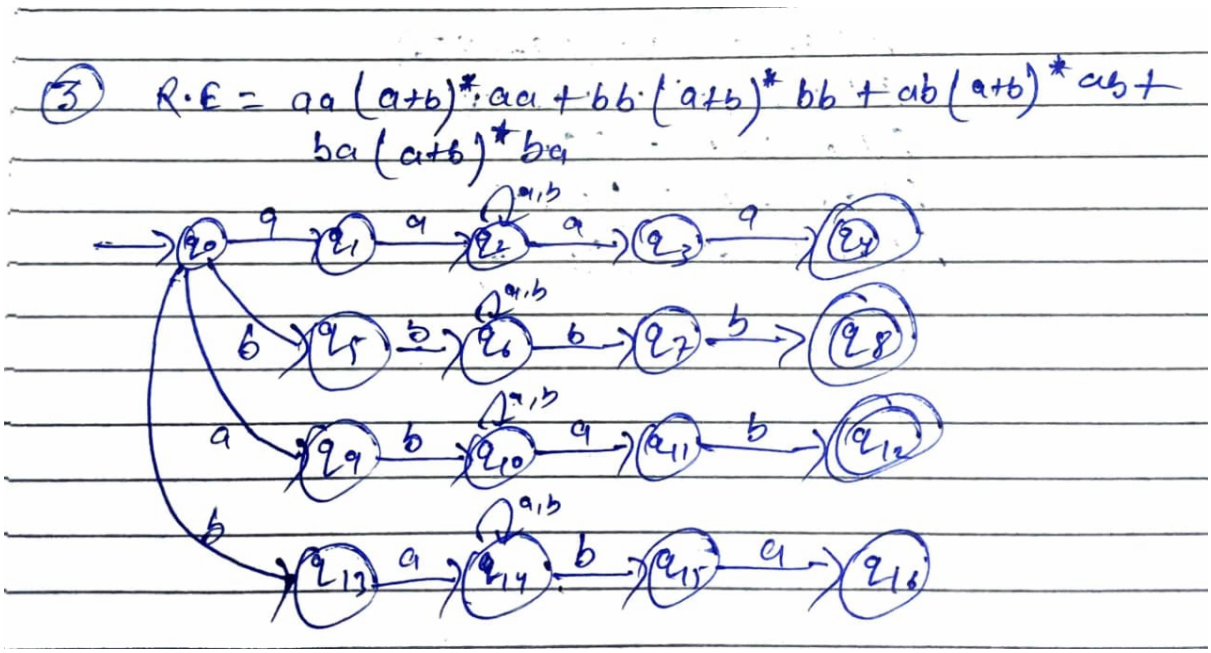
Q3 — First two characters = last two (|w| ≥ 4)

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

void State0(string w, int i, char first, char second);
```

```
void State1(string w, int i, char first, char second);
void State2(string w, int i, char first, char second);
void State3(string w, int i, char first, char second);

int main() {
    string w;
    cout << "Enter a string over {a, b}: ";
    cin >> w;
    if (w.length() < 4) {
        cout << "String is rejected\n";
        return 0;
    }
    State0(w, 0, '\0', '\0');
    return 0;
}

void State0(string w, int i, char first, char second) {
    if (i >= 2) {
        State1(w, i, first, second);
    } else {
        if (i == 0) first = w[i];
        if (i == 1) second = w[i];
        State0(w, i + 1, first, second);
    }
}

void State1(string w, int i, char first, char second) {
    if (i == w.length() - 2) {
        State2(w, i, first, second);
    } else {
        State1(w, i + 1, first, second);
    }
}

void State2(string w, int i, char first, char second) {
    if (w[i] == first && w[i + 1] == second) {
        State3(w, i, first, second);
    } else {
        cout << "String is rejected\n";
    }
}

void State3(string w, int i, char first, char second) {
    cout << "String is accepted\n";
}
```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                                                    Code

PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 3rd.cpp -o 3rd } ; if ($?) { .\3rd }
Enter a string over {a, b}: aabbaa
String is accepted
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 3rd.cpp -o 3rd } ; if ($?) { .\3rd }
Enter a string over {a, b}: aabbbbcd "e:\sem_V\toc_prac\" ; if ($?) { g++ 3rd.cpp -o 3rd } ; if ($?) { .\3rd }
String is rejected
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 3rd.cpp -o 3rd } ; if ($?) { .\3rd }
Enter a string over {a, b}: aaacd "e:\sem_V\toc_prac\" ; if ($?) { g++ 3rd.cpp -o 3rd } ; if ($?) { .\3rd }
String is rejected
PS E:\sem_V\toc_prac>
```



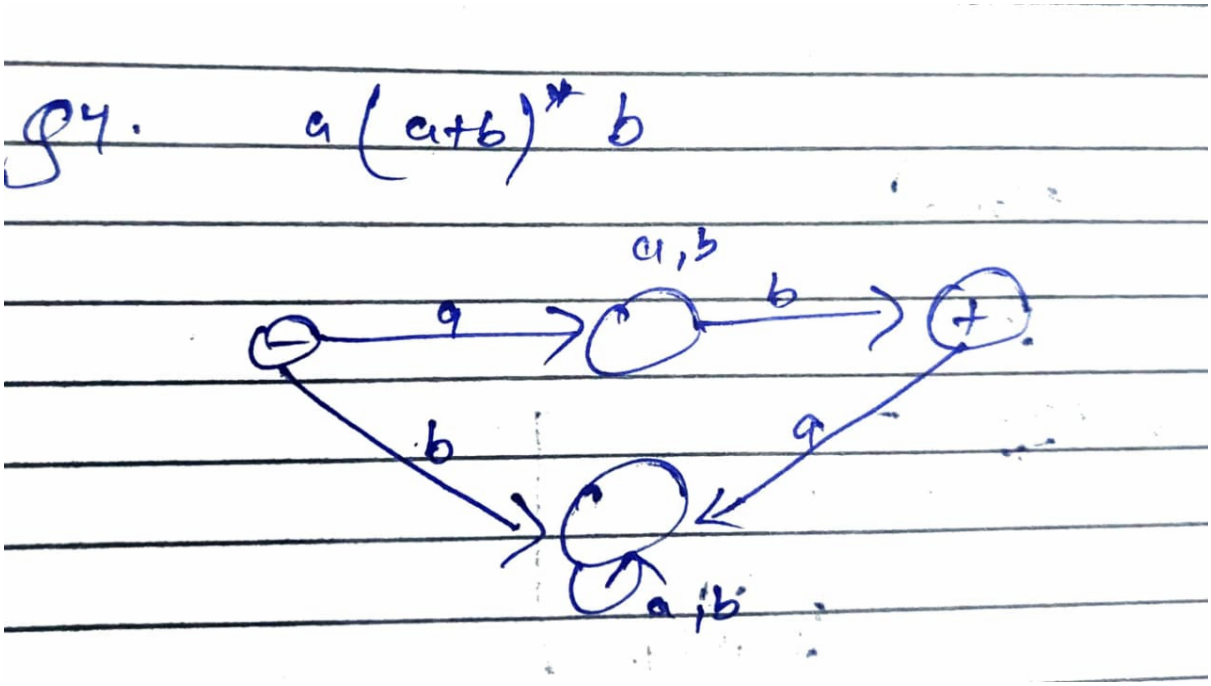
Q4 — L2 = a(a+b)*b

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

void State1(string w, int i);
void State2(string w, int i);
void State3(string w, int i);

int main() {
    string w;
    cout << "Enter string: ";
    cin >> w;
    State1(w, 0);
    return 0;
}

void State1(string w, int i) {
    if (i == w.length()) {
        cout << "String is accepted\n";
```

```
        return;
    }
    if (w[i] == 'a') State2(w, i + 1);
    else if (w[i] == 'b') State1(w, i + 1);
}

void State2(string w, int i) {
    if (i == w.length()) {
        cout << "String is accepted\n";
        return;
    }
    if (w[i] == 'b') State1(w, i + 1);
    else if (w[i] == 'a') State3(w, i + 1);
}

void State3(string w, int i) {
    if (i == w.length()) {
        cout << "String is rejected\n";
        return;
    }
    if (w[i] == 'a' || w[i] == 'b') State3(w, i + 1);
}
```

Output:

```
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 4th.cpp -o 4th } ; if ($?) { .\4th }
Enter string: ab
State 1
State 2
State 1
String is accepted
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 4th.cpp -o 4th } ; if ($?) { .\4th }
Enter string: aa
State 1
```



Q5 — EVEN-EVEN language (even a's, even b's)

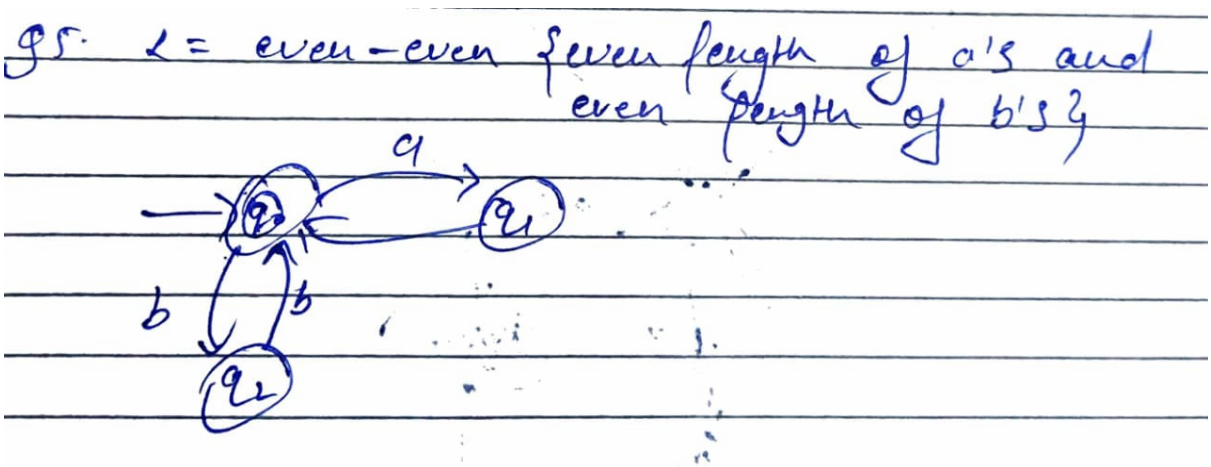
(Fixed, correct version)

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

void State1(string w, int i);
void State2(string w, int i);
void State3(string w, int i);
void State4(string w, int i);

int main() {
    string w;
    cout << "Enter string: ";
    cin >> w;
    State1(w, 0);
    return 0;
}

void State1(string w, int i) { // even a, even b
    if (i == w.length()) { cout<<"Accepted\n"; return; }
    if (w[i]=='a') State2(w,i+1);
    else State3(w,i+1);
}
```

```
void State2(string w, int i) { // odd a, even b
    if (i == w.length()) { cout<<"Rejected\n"; return; }
    if (w[i]=='a') State1(w,i+1);
    else State4(w,i+1);
}

void State3(string w, int i) { // even a, odd b
    if (i == w.length()) { cout<<"Rejected\n"; return; }
    if (w[i]=='a') State4(w,i+1);
    else State1(w,i+1);
}

void State4(string w, int i) { // odd a, odd b
    if (i == w.length()) { cout<<"Rejected\n"; return; }
    if (w[i]=='a') State3(w,i+1);
    else State2(w,i+1);
}
```

Output:

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 5th.cpp -o 5th } ; if ($?) { .\5th }
Enter string over {a,b}: a
State 1
State 2
String is rejected
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 5th.cpp -o 5th } ; if ($?) { .\5th }
Enter string over {a,b}: aabb
State 1
State 2
State 1
State 3
State 1
String is accepted
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 5th.cpp -o 5th } ; if ($?) { .\5th }
Enter string over {a,b}: aa
State 1
```



Q6 — Union, Intersection, Concatenation

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

int unionLanguage(char* w) {
    int len = strlen(w);
    if (w[0]=='a') return 1;
    if (w[len-1]=='b') return 1;
    return 0;
}

int intersectionLanguage(char* w) {
    int len = strlen(w);
    if (w[0]=='a' && w[len-1]=='b') return 1;
    return 0;
}

int concatenationLanguage(char* w) {
    int len = strlen(w);
    if (w[0]=='a' && w[len-1]=='b') return 1;
    return 0;
}

int main() {
    char w[100];
    cout<<"Enter string: ";
    cin>>w;

    cout<<"Union: "<<(unionLanguage(w)?"Accepted":"Rejected")<<"\n";
    cout<<"Intersection: "<<(intersectionLanguage(w)?"Accepted":"Rejected")<<"\n";
    cout<<"Concatenation: "<<(concatenationLanguage(w)?"Accepted":"Rejected")<<"\n";
    return 0;
}
```

Output:

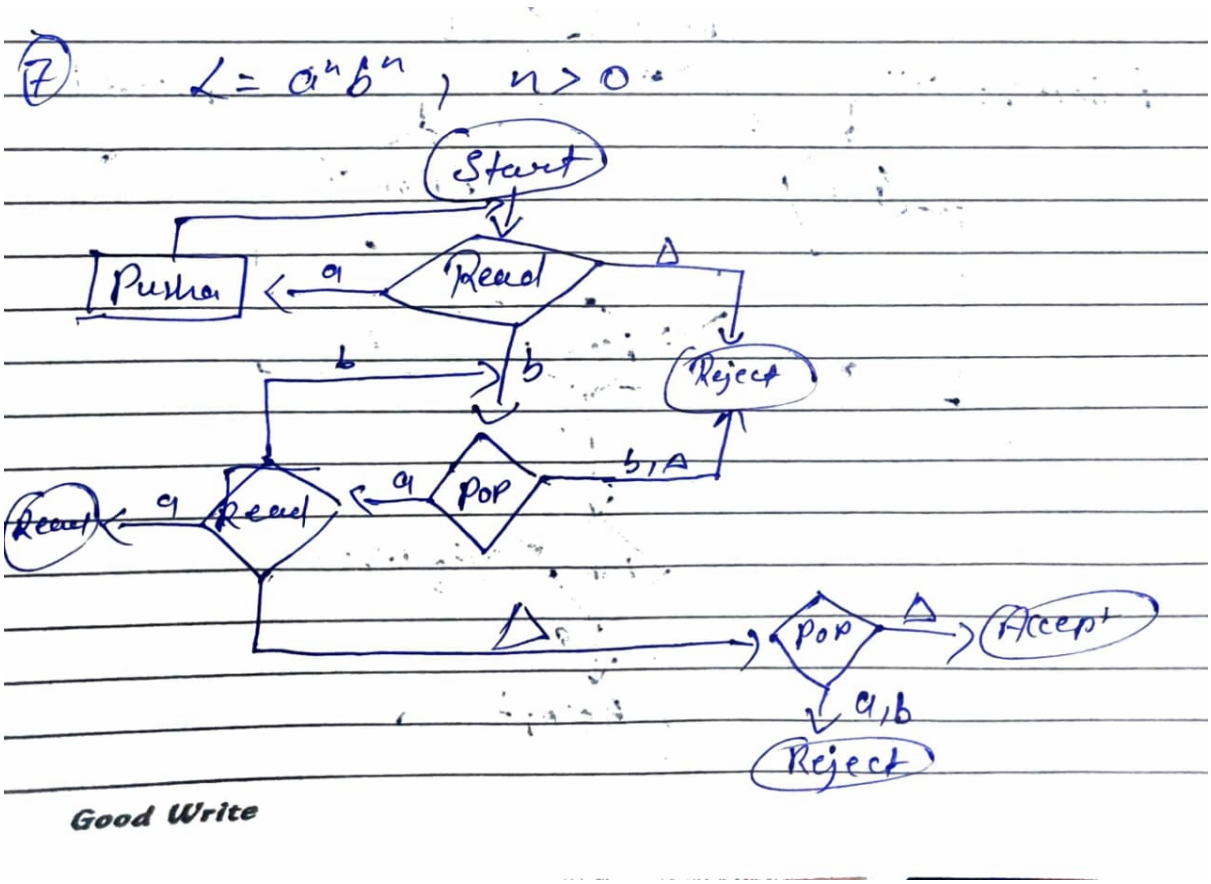
```
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 6th.cpp -o 6th } ; if ($?) { .\6th }
Enter string: a0
Union Language Result: Accepted
Intersection Language Result: Rejected
Concatenation Language Result: Rejected
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 6th.cpp -o 6th } ; if ($?) { .\6th }
Enter string: aaab
Union Language Result: Accepted
Intersection Language Result: Accepted
Concatenation Language Result: Accepted
PS E:\sem_V\toc_prac>
```

Q7 — PDA for $a^n b^n$

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

int simulatePDA(char* input) {
    stack<char> s;
    int len = strlen(input);

    for(int i=0;i<len;i++){
        if(input[i]=='a') s.push('a');
        else if(input[i]=='b'){
            if(s.empty()) return 0;
            s.pop();
        }
        else return 0;
    }
    return s.empty();
}

int main() {
    char input[100];
    cout<<"Enter string: ";
    cin>>input;
    cout<<(simulatePDA(input)?"Accepted":"Rejected")<<"\n";
    return 0;
}
```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 7th.cpp -o 7th } ; if ($?) { .\7th }
Enter string: ab
String accepted
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 7th.cpp -o 7th } ; if ($?) { .\7th }
Enter string: aaaabbbbcd "e:\sem_V\toc_prac\" ; if ($?) { g++ 7th.cpp -o 7th } ; if ($?) { .\7th }
String rejected
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 7th.cpp -o 7th } ; if ($?) { .\7th }
Enter string: aaaabbbb
String accepted
PS E:\sem_V\toc_prac>
```



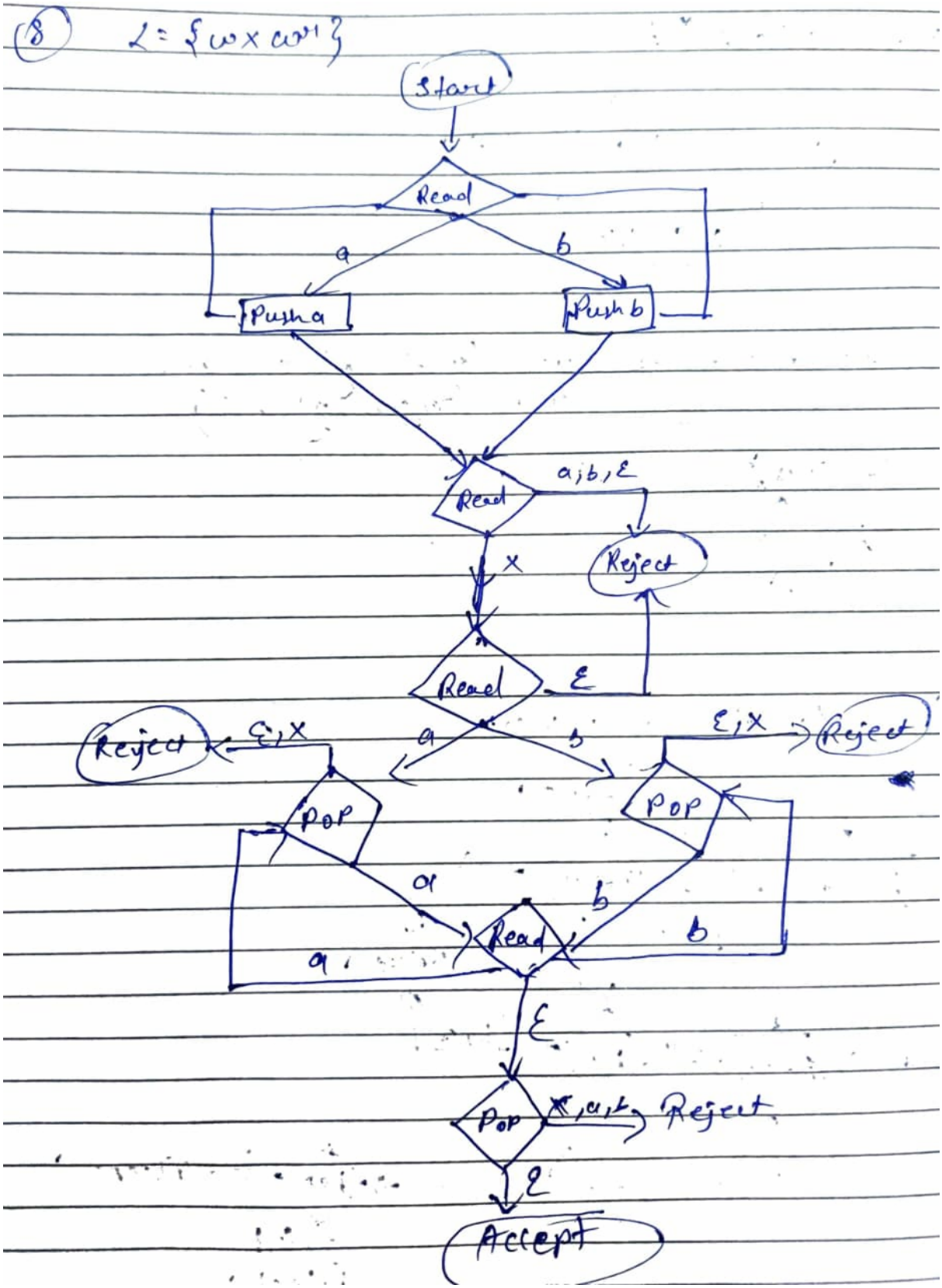
Q8 — PDA for w X w^r

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

bool simulate(string w) {
    stack<char> s;
    bool seenX = false;

    for(char c : w){
        if(!seenX){
```

```
            if(c=='a' || c=='b') s.push(c);
            else if(c=='X') seenX=true;
            else return false;
        }
        else{
            if(s.empty() || s.top()!=c) return false;
            s.pop();
        }
    }
    return seenX && s.empty();
}

int main() {
    string w;
    cout<<"Enter string: ";
    cin>>w;
    cout<<(simulate(w)?"Accepted":"Rejected")<<"\n";
    return 0;
}
```

Output:

```
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 8th.cpp -o 8th } ; if ($?) { .\8th }
Enter string over {a,b,X} (form wXw^r): aaaXaaa
String accepted
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 8th.cpp -o 8th } ; if ($?) { .\8th }
Enter string over {a,b,X} (form wXw^r): aX
String rejected
```



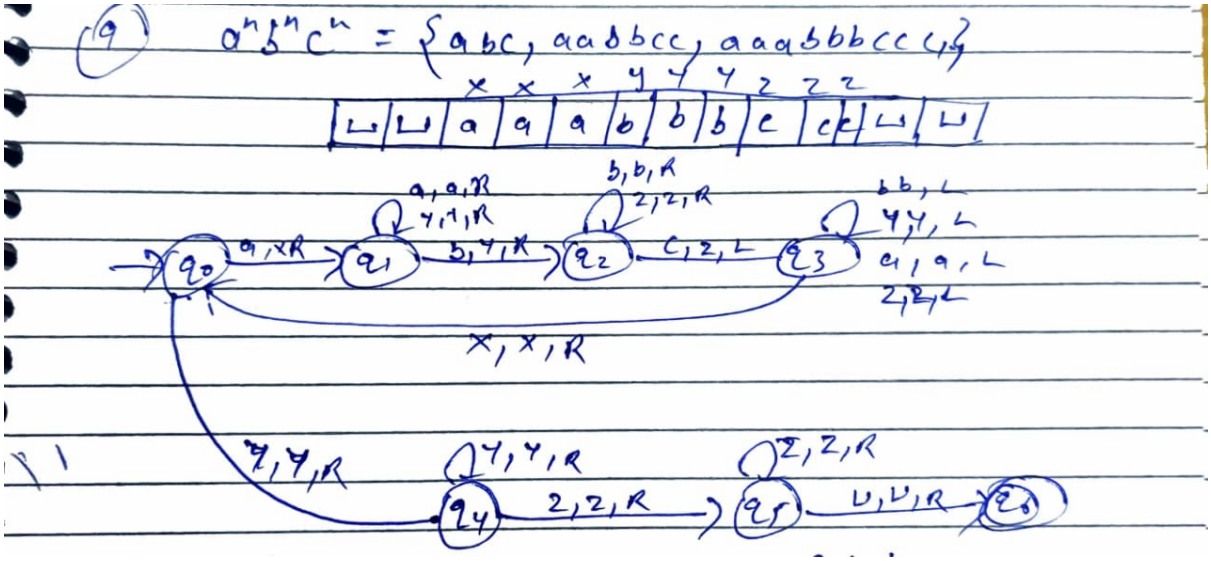
Q9 — TM for $a^n b^n c^n$ (corrected)

```
#include <iostream>
```

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

int simulate(char* input) {
    int len=strlen(input);
    int i=0,a=0,b=0,c=0;

    while(i<len && input[i]=='a'){a++; i++;}
    while(i<len && input[i]=='b'){b++; i++;}
    while(i<len && input[i]=='c'){c++; i++;}

    if(i!=len) return 0;
    if(a==0 || b==0 || c==0) return 0;
    return (a==b && b==c);
}

int main() {
    char w[100];
    cout<<"Enter string: ";
    cin>>w;
    cout<<(simulate(w)?"Accepted":"Rejected")<<"\n";
    return 0;
}
```

Output:

```
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 9th.cpp -o 9th } ; if ($?) { .\9th }
Enter string over {a,b,c}: aabbcc
String accepted
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 9th.cpp -o 9th } ; if ($?) { .\9th }
Enter string over {a,b,c}: aabbc
String rejected
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 9th.cpp -o 9th } ; if ($?) { .\9th }
Enter string over {a,b,c}: aaccbb
String rejected
PS E:\sem_V\toc_prac>
```



Q10 — TM for incrementing binary number

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

void increment(char* in) {
    int len=strlen(in);
    int carry=1;

    for(int i=len-1;i>=0;i--){
        if(in[i]=='0'){ in[i]='1'; carry=0; break; }
        else if(in[i]=='1') in[i]='0';
        else return;
    }

    if(carry){
        memmove(in+1,in,len+1);
        in[0]='1';
    }
}

int main() {
    char w[100];
    cout<<"Enter binary number: ";
    cin>>w;
    increment(w);
    cout<<"Incremented: "<<w<<"\n";
    return 0;
```

```
}
```

Output:

```
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 10th.cpp -o 10th } ; if ($?) { .\10th }
Enter binary number: 100
Incremented number: 101
PS E:\sem_V\toc_prac> cd "e:\sem_V\toc_prac\" ; if ($?) { g++ 10th.cpp -o 10th } ; if ($?) { .\10th }
Enter binary number: 111
Incremented number: 1000
PS E:\sem_V\toc_prac>
```