**Practical : 8**

**Topic :**

Write a program to implement the logic gates functionality for the following: AND gate, OR gate, NOT gate, NAND gate, NOR gate, XOR gate and XNOR gate. After creating a successful script for the gates' implementation, you need to superimpose the implementation for prediction of a color on the basis of combination of three colors using logic gates

**Implementation :**

1. Logic Gates

#include <bits/stdc++.h>

using namespace std;

void INPUT();

class Operations

{

bool n1, n2, ans;

public:

void inp() {

cout << "\nEnter two bits as input :\n";

cin >> n1 >> n2;

}

void Oper() {

cout << "\n\*\* Operations \*\*";

cout << "\n1. AND :" << (n1 & n2);

cout << "\n2. OR :" << (n1 | n2);

cout << "\n3. XOR :" << (n1 ^ n2);

cout << "\n4. NAND :" << !(n1 & n2);

cout << "\n5. NOR :" << !(n1 & n2);

cout << "\n6. NOT :" << !n1;

//cout << "\n7. XNOR :"<<(n1 == n2); // same bits ==> 1 ,

cout << "\n7. XNOR :" << ((n1 | !n2) & (!n1 + n2)) << "\n";

}

};

int main()

{

int choice = -1;

Operations op;

do

{

cout << "\n\*\*\*\* MENU \*\*\*\*\n";

cout << "\n0. Exit";

cout << "\n1. Input numbers";

cout << "\n2. Perform Operations\n";

cin >> choice;

switch (choice) {

case 1 :

op.inp(); break;

case 2 :

op.Oper(); break;

default :

cout << "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n"; break;

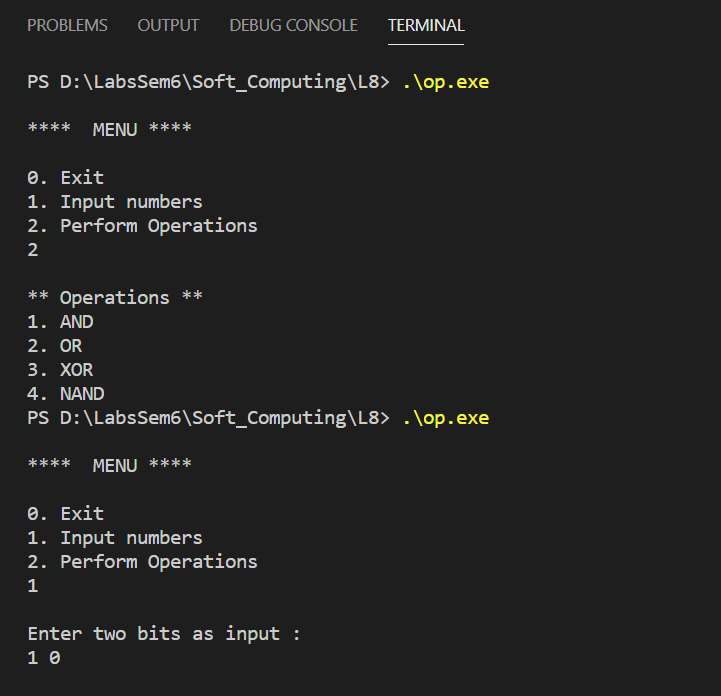
}

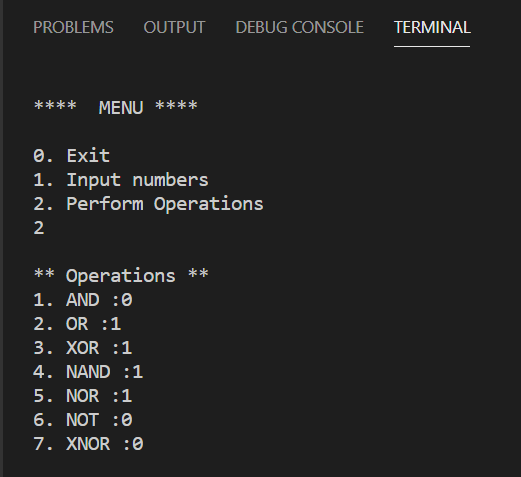
} while (choice != 0);

return 0;

}

Output :





1. Color prediction

#include <bits/stdc++.h>

using namespace std;

struct colors

{

map<string, vector<int>> composition;

bool exnor(int a, int b) {

// if we do bitwise xnor,

//output is 1 if both bits are same => equivalent to a == b

return a == b;

}

void init() {

//red, yellow, blue

//primary

composition["Red"] = {4, 0, 0};

composition["Yellow"] = {0, 4, 0};

composition["Blue"] = {0, 0, 4};

//secondary

composition["Orange"] = {2, 2, 0};

composition["Voilet"] = {2, 0, 2};

composition["Green"] = {0, 2, 2};

//tertariary

composition["Yellow Orange"] = {1, 3, 0};

composition["Red Orange"] = {3, 1, 0};

composition["Red Voilet"] = {3, 0, 1};

composition["Blue Voilet"] = {1, 0, 3};

composition["Blue Green"] = {0, 1, 3};

composition["Yellow Green"] = {0, 3, 1};

}

string col(int i) {

switch (i) {

case 1: return "Red";

case 2: return "Yellow";

case 3: return "Blue";

case 4: return "Orange";

case 5: return "Voilet";

case 6: return "Green";

case 7: return "Yellow Orange";

case 8: return "Red Orange";

case 9: return "Red Voilet";

case 10: return "Blue Voilet";

case 11: return "Blue Green";

case 12: break;

default : break;

}

return "Yellow Green";

}

string dis(vector<int> c) {

float d = 10000;

string res = "";

for (auto i : composition) {

float t = 0;

int it = 0;

for (int j : i.second) {

t += pow(c[it] - j, 2);

it++;

}

t = sqrt(t);

if (t < d) {

res = i.first;

d = t;

}

}

return res;

}

string predict(string c1, string c2) {

auto m1 = composition[c1];

auto m2 = composition[c2];

bool unexpected = false;

for (int i = 0; i < 3; ++i)

{

m1[i] = (m1[i] + m2[i]) / 2;

}

int type = 4;

for (int i : m1) {

if (i)

type = min(type, i);

}

if (type == 4) {

// 1 out of 3 primary coolors

if (exnor(m1[0], 4) and exnor(m1[1], 0) and exnor(m1[2], 0))

return "Red";

else if (exnor(m1[0], 0) and exnor(m1[1], 4) and exnor(m1[2], 0))

return "Yellow";

else if (exnor(m1[0], 0) and exnor(m1[1], 0) and exnor(m1[2], 4))

return "Blue";

else

unexpected = true;

} else if (type == 2) {

//one of secondary color

if (exnor(m1[0], 2) and exnor(m1[1], 2) and exnor(m1[2], 0))

return "Orange";

else if (exnor(m1[0], 0) and exnor(m1[1], 2) and exnor(m1[2], 2))

return "Green";

else if (exnor(m1[0], 2) and exnor(m1[1], 0) and exnor(m1[2], 2))

return "Voilet";

else

unexpected = true;

} else if (type == 1) {

if (exnor(m1[0], 1) and exnor(m1[2], 0))

return "Yellow Orange";

else if (exnor(m1[0], 1) and exnor(m1[2], 0))

return "Red Orange";

else if (exnor(m1[1], 0) and exnor(m1[2], 1))

return "Red Voilet";

else if (exnor(m1[1], 1) and exnor(m1[2], 0))

return "Blue Voilet";

else if (exnor(m1[0], 0) and exnor(m1[1], 1))

return "Blue Green";

else if (exnor(m1[0], 0) and exnor(m1[2], 1))

return "Yellow Green";

else

unexpected = true;

}

string res = c1;

if (unexpected) {

res = dis(m1);

}

return res;

}

};

int main()

{

colors c;

c.init();

cout << "Enter Two numbers corresponding to colors from following\n";

cout << "1. Red \n";

cout << "2. Yellow\n";

cout << "3. Blue \n";

cout << "4. Orange\n";

cout << "5. Voilet\n";

cout << "6. Green\n";

cout << "7. Yellow Orange\n";

cout << "8. Red Orange\n";

cout << "9. Red Voilet \n";

cout << "10. Blue Voilet \n";

cout << "11. Blue Green \n";

cout << "12. Yellow Green \n";

int c1, c2;

cin >> c1 >> c2;

cout << "Resultant color :" << c.predict(c.col(c1), c.col(c2));

return 0;

}

Output :

