**Program:**

import java.util.\*;

class Environment {

Scanner scr = new Scanner(System.in);

int np;

int wp, gp;

int pos[];

int b\_pos[] = new int[20];

int s\_pos[] = new int[20];

void accept(String w[][]) {

for (int i = 0; i < 20; ++i) {

b\_pos[i] = -1;

s\_pos[i] = -1;

}

for (int i = 0; i < 5; ++i)

for (int j = 0; j < 5; ++j)

w[i][j] = "";

int count = 1;

System.out.println("The positions are as follows.");

for (int i = 1; i <= 4; ++i) {

System.out.println("\n-----------------------------------------------------------------");

System.out.print("|\t");

for (int j = 1; j <= 4; ++j)

System.out.print((count++) + "\t|\t");

}

System.out.println("\n-----------------------------------------------------------------");

System.out.println("\nAgent start position: 13");

w[4][1] = "A";

System.out.println("\nEnter the number of pits.");

np = scr.nextInt();

pos = new int[np];

System.out.println("Positions of pit, gold and wumpus should not overlap.");

System.out.println("Enter the position of pits.");

for (int i = 0; i < np; ++i) {

pos[i] = scr.nextInt();

show\_sense(pos[i], 1, w);

}

System.out.println("Enter the position of wumpus.");

wp = scr.nextInt();

show\_sense(wp, 2, w);

System.out.println("Enter the position of gold.");

gp = scr.nextInt();

insert(w);

}

void insert(String w[][]) {

int temp = 0;

int count = 0;

int flag1 = 0, flag2 = 0;

for (int i = 0; i < np; ++i) {

temp = pos[i];

count = 0;

for (int j = 1; j <= 4; ++j) {

for (int k = 1; k <= 4; ++k) {

++count;

if (count == temp)

w[j][k] += "P";

else if (count == gp && flag1 == 0) {

w[j][k] += "G";

flag1 = 1;

} else if (count == wp && flag2 == 0) {

w[j][k] += "W";

flag2 = 1;

}

}

}

}

display(w);

}

void show\_sense(int a, int b, String w[][]) {

int t1, t2, t3, t4;

t1 = a - 1;

t2 = a + 1;

t3 = a + 4;

t4 = a - 4;

if (a == 5 || a == 9)

t1 = 0;

if (a == 8 || a == 12)

t2 = 0;

if (a == 4)

t2 = 0;

if (a == 13)

t1 = 0;

if (t3 > 16)

t3 = 0;

if (t4 < 0)

t4 = 0;

if (b == 1)

{b\_pos[0] = t1; b\_pos[1] = t2; b\_pos[2] = t3; b\_pos[3] = t4;}

else if (b == 2)

{s\_pos[0] = t1; s\_pos[1] = t2; s\_pos[2] = t3; s\_pos[3] = t4;}

int temp1, count;

for (int i = 0; i < 4; ++i) {

if (b == 1)

temp1 = b\_pos[i];

else

temp1 = s\_pos[i];

count = 0;

for (int j = 1; j <= 4; ++j) {

for (int k = 1; k <= 4; ++k) {

++count;

if (count == temp1 && b == 1 && !w[j][k].contains("B")) {

w[j][k] += "B";

} else if (count == temp1 && b == 2 && !w[j][k].contains("S"))

w[j][k] += "S";

}

}

}

}

void display(String w[][]) {

System.out.println("\nThe environment for problem is as follows.\n");

for (int i = 1; i <= 4; ++i) {

System.out.println("\n-----------------------------------------------------------------");

System.out.print("|\t");

for (int j = 1; j <= 4; ++j)

System.out.print(w[i][j] + "\t|\t");

}

System.out.println("\n-----------------------------------------------------------------");

}

}

class tiles {

int safe = 0;

int unsafe = 0;

int wump = 0;

int pit = 0;

int gold = 0;

int doubt\_pit = 0;

int doubt\_wump = 0;

String env;

int num = 0;

int br = 0;

int bl = 0;

int bu = 0;

int bd = 0;

int visited = 0;

int l, r, u, d;

String back = "";

tiles(String s, int n) {

env = s;

num = n;

l = r = u = d = 0;

if (n == 9 || n == 5)

bl = 1;

if (n == 8 || n == 12)

br = 1;

if (n == 1) {

bu = 1; bl = 1;

}

if (n == 13) {

bd = 1; bl = 1;

}

if (n == 4) {

bu = 1; br = 1;

}

if (n == 16) {

bd = 1; br = 1;

}

}

int sense() {

if (env.contains("B"))

return 1;

else if (env.contains("S"))

return 2;

else if (env.contains("G"))

return 3;

if (env.contains("W"))

return 4;

else

return 0;

}

}

class wumpus {

static int scream = 0;

static int score = 0;

static int complete = 0;

static boolean check(tiles t) {

int temp = t.sense();

if (temp == 1 || temp == 2)

return false;

return true;

}

public static void main(String args[]) {

Scanner scr = new Scanner(System.in);

Environment e = new Environment();

String w[][] = new String[5][5];

e.accept(w);

System.out.println("\n\nFinding the solution...");

tiles t[] = new tiles[17];

int c = 1;

out: for (int i = 1; i < 5; ++i) {

for (int j = 1; j < 5; ++j) {

if (c > 16)

break out;

t[c] = new tiles(w[i][j], c);

++c;

}

}

t[13].safe = 1;

t[13].visited = 1;

int pos = 13;

int condition;

int limit = 0;

String temp1, temp2;

do {

++limit;

condition = -1;

if (t[pos].env.contains("G")) {

complete = 1;

System.out.println("Gold Found!!");

break;

}

if (t[pos].br != 1 && t[pos].r != 1 && t[pos + 1].doubt\_pit < 1 && t[pos + 1].doubt\_wump < 1 && t[pos + 1].pit != 1 && t[pos + 1].wump != 1 && !(t[pos].back.contains("r") && (t[pos].l != 1 || t[pos].u != 1 || t[pos].d != 1) && check(t[pos]) )) {

temp1 = "l";

t[pos].r = 1;

++pos;

System.out.println("\nfront pos=" + pos);

++score;

t[pos].back += temp1;

condition = t[pos].sense();

if (condition == 3)

{complete = 1; break;}

else if (condition == 1 && t[pos].visited == 0) {

if (t[pos].br != 1 && t[pos + 1].safe != 1)

t[pos + 1].doubt\_pit += 1;

if (t[pos].bu != 1 && (pos - 4) >= 1 && t[pos - 4].safe != 1)

t[pos - 4].doubt\_pit += 1;

if (t[pos].bl != 1 && t[pos - 1].safe != 1 )

t[pos - 1].doubt\_pit += 1;

if (t[pos].bd != 1 && (pos + 4) <= 16 && t[pos + 4].safe != 1)

t[pos + 4].doubt\_pit += 1;

t[pos].safe = 1;

} else if (condition == 2 && t[pos].visited == 0) {

if (t[pos].br != 1 && t[pos + 1].safe != 1)

t[pos + 1].doubt\_wump += 1;

if (t[pos].bu != 1 && (pos - 4) >= 1 && t[pos - 4].safe != 1)

t[pos - 4].doubt\_wump += 1;

if (t[pos].bl != 1 && t[pos - 1].safe != 1)

t[pos - 1].doubt\_wump += 1;

if (t[pos].bd != 1 && (pos + 4) <= 16 && t[pos + 4].safe != 1)

t[pos + 4].doubt\_wump += 1;

t[pos].safe = 1;

}else if (condition == 0)

t[pos].safe = 1;

t[pos].visited = 1;

} else if (t[pos].bl != 1 && t[pos].l != 1 && t[pos - 1].doubt\_pit < 1 && t[pos - 1].doubt\_wump < 1 && t[pos - 1].pit != 1 && t[pos - 1].wump != 1 && !(t[pos].back.contains("l") && (t[pos].r != 1 || t[pos].u != 1 || t[pos].d != 1) && check(t[pos]) )) {

temp1 = "r";

t[pos].l = 1;

pos = pos - 1;

System.out.println("\nback pos= " + pos);

++score;

t[pos].back += temp1;

condition = t[pos].sense();

if (condition == 3)

{complete = 1; break;}

else if (condition == 1 && t[pos].visited == 0) {

if (t[pos].br != 1 && t[pos + 1].safe != 1)

t[pos + 1].doubt\_pit += 1;

if (t[pos].bu != 1 && (pos - 4) >= 1 && t[pos - 4].safe != 1)

t[pos - 4].doubt\_pit += 1;

if (t[pos].bl != 1 && t[pos - 1].safe != 1)

t[pos - 1].doubt\_pit += 1;

if (t[pos].bd != 1 && (pos + 4) <= 16 && t[pos + 4].safe != 1)

t[pos + 4].doubt\_pit += 1;

t[pos].safe = 1;

} else if (condition == 2 && t[pos].visited == 0) {

if (t[pos].br != 1 && t[pos + 1].safe != 1)

t[pos + 1].doubt\_wump += 1;

if (t[pos].bu != 1 && (pos - 4) >= 1 && t[pos - 4].safe != 1)

t[pos - 4].doubt\_wump += 1;

if (t[pos].bl != 1 && t[pos - 1].safe != 1)

t[pos - 1].doubt\_wump += 1;

if (t[pos].bd != 1 && (pos + 4) <= 16 && t[pos + 4].safe != 1)

t[pos + 4].doubt\_wump += 1;

t[pos].safe = 1;

} else if (condition == 0)

t[pos].safe = 1;

t[pos].visited = 1;

} else if (t[pos].bu != 1 && t[pos].u != 1 && (pos - 4) >= 1 && t[pos - 4].doubt\_pit < 1 && t[pos - 4].doubt\_wump < 1 && t[pos - 4].pit != 1 && t[pos - 1].wump != 1 && !(t[pos].back.contains("u") && (t[pos].l != 1 || t[pos].r != 1 || t[pos].d != 1) && check(t[pos]) )) {

temp1 = "d";

t[pos].u = 1;

pos = pos - 4;

System.out.println("\nUp pos= " + pos);

++score;

t[pos].back += temp1;

condition = t[pos].sense();

if (condition == 3)

{complete = 1; break;}

else if (condition == 1 && t[pos].visited == 0) {

if (t[pos].br != 1 && t[pos + 1].safe != 1)

t[pos + 1].doubt\_pit += 1;

if (t[pos].bu != 1 && (pos - 4) >= 1 && t[pos - 4].safe != 1)

t[pos - 4].doubt\_pit += 1;

if (t[pos].bl != 1 && t[pos - 1].safe != 1)

t[pos - 1].doubt\_pit += 1;

if (t[pos].bd != 1 && (pos + 4) <= 16 && t[pos + 4].safe != 1)

t[pos + 4].doubt\_pit += 1;

t[pos].safe = 1;

} else if (condition == 2 && t[pos].visited == 0) {

if (t[pos].br != 1 && t[pos + 1].safe != 1)

t[pos + 1].doubt\_wump += 1;

if (t[pos].bu != 1 && (pos - 4) >= 1 && t[pos - 4].safe != 1)

t[pos - 4].doubt\_wump += 1;

if (t[pos].bl != 1 && t[pos - 1].safe != 1)

t[pos - 1].doubt\_wump += 1;

if (t[pos].bd != 1 && (pos + 4) <= 16 && t[pos + 4].safe != 1)

t[pos + 4].doubt\_wump += 1;

t[pos].safe = 1;

} else if (condition == 0)

t[pos].safe = 1;

t[pos].visited = 1;

} else if (t[pos].bd != 1 && t[pos].d != 1 && (pos + 4) <= 16 && t[pos + 4].doubt\_pit < 1 && t[pos + 4].doubt\_wump < 1 && t[pos + 4].pit != 1 && t[pos + 4].wump != 1) {

temp1 = "u";

t[pos].d = 1;

pos = pos + 4;

System.out.println("\ndown pos= " + pos);

++score;

t[pos].back += temp1;

condition = t[pos].sense();

if (condition == 3)

{complete = 1; break;}

else if (condition == 1 && t[pos].visited == 0) {

if (t[pos].br != 1 && t[pos + 1].safe != 1)

t[pos + 1].doubt\_pit += 1;

if (t[pos].bu != 1 && (pos - 4) >= 1 && t[pos - 4].safe != 1)

t[pos - 4].doubt\_pit += 1;

if (t[pos].bl != 1 && t[pos - 1].safe != 1)

t[pos - 1].doubt\_pit += 1;

if (t[pos].bd != 1 && (pos + 4) <= 16 && t[pos + 4].safe != 1)

t[pos + 4].doubt\_pit += 1;

t[pos].safe = 1;

} else if (condition == 2 && t[pos].visited == 0) {

if (t[pos].br != 1 && t[pos + 1].safe != 1)

t[pos + 1].doubt\_wump += 1;

if (t[pos].bu != 1 && (pos - 4) >= 1 && t[pos - 4].safe != 1)

t[pos - 4].doubt\_wump += 1;

if (t[pos].bl != 1 && t[pos - 1].safe != 1)

t[pos - 1].doubt\_wump += 1;

if (t[pos].bd != 1 && (pos + 4) <= 16 && t[pos + 4].safe != 1)

t[pos + 4].doubt\_wump += 1;

t[pos].safe = 1;

} else if (condition == 0)

t[pos].safe = 1;

t[pos].visited = 1;

} else if (limit > 50) {

int temp3 = pos;

int flag\_1 = 0, flag2 = 0, flag3 = 0, flag4 = 0;

System.out.println("\nCurrently at position " + temp3 + ".\nThinking....");

while (t[pos].visited == 1 && t[pos].br != 1) {

++pos;

++score;

}

if (t[pos].pit == 1 || t[pos].wump == 1 || (t[pos].br == 1 && t[pos].visited == 1 && t[pos].safe != 1)) {

pos = temp3;

flag\_1 = 1;

}

if (flag\_1 == 0)

t[pos].back += "l";

while (pos + 4 >= 1 && t[pos].bu != 1 && t[pos].visited == 1) {

pos -= 4;

++score;

}

if (t[pos].pit == 1 || t[pos].wump == 1 || (t[pos].bu == 1 && t[pos].visited == 1 && t[pos].safe != 1)) {

pos = temp3;

flag3 = 1;

}

if (flag3 == 0)

t[pos].back += "d";

while (t[pos].visited == 1 && t[pos].bl != 1) {

--pos;

++score;

}

if (t[pos].pit == 1 || t[pos].wump == 1 || (t[pos].bl == 1 && t[pos].visited == 1 && t[pos].safe != 1)) {

pos = temp3;

flag2 = 1;

}

if (flag2 == 0)

t[pos].back += "r";

while (pos + 4 <= 16 && t[pos].bd != 1 && t[pos].visited == 1) {

pos += 4;

++score;

}

if (t[pos].pit == 1 || t[pos].wump == 1 || (t[pos].bd == 1 && t[pos].visited == 1 && t[pos].safe != 1)) {

pos = temp3;

flag4 = 1;

}

if (flag4 == 0)

t[pos].back += "u";

t[pos].safe = 1;

t[pos].visited = 1;

System.out.println("reached at position " + pos);

limit = 0;

}

if (t[pos].env.contains("W") && scream != 1) {

score += 100;

scream = 1;

t[pos].safe = 1;

System.out.println("\n\nWumpus killed >--0-->");

t[pos].env.replace("W", " ");

for (int l = 1; l <= 16; ++l) {

t[l].doubt\_wump = 0;

t[l].env.replace("S", " ");

}

}

if (t[pos].env.contains("P")) {

score += 50;

t[pos].pit = 1;

System.out.println("\n\nFallen in pit of position " + pos + ".");

}

for (int k = 1; k <= 16; ++k) {

if (t[k].doubt\_pit == 1 && t[k].doubt\_wump == 1) {

t[k].doubt\_pit = 0;

t[k].doubt\_wump = 0;

t[k].safe = 1;

}

}

for (int y = 1; y <= 16; ++y) {

if (t[y].doubt\_wump > 1) {

t[y].wump = 1;

for (int h = 1; h <= 16; ++h) {

if (h != y) {

t[h].doubt\_wump = 0;

t[h].env.replace("S", " ");

}

}

}

}

for (int y = 1; y <= 16; ++y) {

if (t[y].doubt\_pit > 1) {

t[y].pit = 1;

}

}

try {Thread.sleep(200);} catch (Exception p) {}

} while (complete == 0);

if (complete == 1) {

score \*= -1;

score += 1000;

}

System.out.println("The score of the agent till he reaches gold is " + score + ".\n Now he will return back following the best explored path.");

}

}

**Output:**



