#include <bits/stdc++.h>

using namespace std;

string ltrim(const string &);

string rtrim(const string &);

vector<string> split(const string &);

int hourglassSum(vector<vector<int>> arr) {

int max\_sum = INT\_MIN;

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

for (int j = 0; j <= 3; j++) {

int sum = arr[i][j] + arr[i][j + 1] + arr[i][j + 2] +

arr[i + 1][j + 1] +

arr[i + 2][j] + arr[i + 2][j + 1] + arr[i + 2][j + 2];

if (sum > max\_sum) {

max\_sum = sum;

}

}

}

return max\_sum;

}

int main()

{

ofstream fout(getenv("OUTPUT\_PATH"));

vector<vector<int>> arr(6);

for (int i = 0; i < 6; i++) {

arr[i].resize(6);

string arr\_row\_temp\_temp;

getline(cin, arr\_row\_temp\_temp);

vector<string> arr\_row\_temp = split(rtrim(arr\_row\_temp\_temp));

for (int j = 0; j < 6; j++) {

int arr\_row\_item = stoi(arr\_row\_temp[j]);

arr[i][j] = arr\_row\_item;

}

}

int result = hourglassSum(arr);

fout << result << "\n";

fout.close();

return 0;

}

string ltrim(const string &str) {

string s(str);

s.erase(

s.begin(),

find\_if(s.begin(), s.end(), not1(ptr\_fun<int, int>(isspace)))

);

return s;

}

string rtrim(const string &str) {

string s(str);

s.erase(

find\_if(s.rbegin(), s.rend(), not1(ptr\_fun<int, int>(isspace))).base(),

s.end()

);

return s;

}

vector<string> split(const string &str) {

vector<string> tokens;

string::size\_type start = 0;

string::size\_type end = 0;

while ((end = str.find(" ", start)) != string::npos) {

tokens.push\_back(str.substr(start, end - start));

start = end + 1;

}

tokens.push\_back(str.substr(start));

return tokens;

}