

Forum:[Hack Forums](#)

Thread: [Update on Minesweeper project](#)

shaggy756 - April 23, 2014, 12:24 pm

If you saw my thread earlier, you know that I'm currently working on a text based minesweeper program in C++. I've got a lot of the base game down but I need assistance with a few bugs and mechanics that I don't really understand.

Things I need help on: How to hide the mines (0 is a free space, 1 is a mine) behind a symbol, like a +, while keeping the value equivalent (1 or 0)

What seems to be a bug where I enter the coordinates, it runs through the if statements, and then changes the number of a different coordinate (you will probably have to run the program to see what I mean)

If anybody could throw in their input on what I'm missing, or would be willing to help guide me through over Skype, I'd greatly appreciate it. Really trying to get a better understanding of the language.

Here's my code so far:

```
Code #include <iostream>
#include <cstdlib>
```

```
using namespace std;
```

```
int main(){
    int mines = 0, x, y, i, j, mark = 0, repeat;
    int grid[10][10] = {0}; //This sets whole array to zeroes

    //----- Random mine placement

    cout<<"Placing Mines..."<<endl;
    do{
        do{
            x = rand()%10; //get a random row
            y = rand()%10; //get a random column
        }while (grid[x][y] != 0); //do again if a mine is located there already
        grid[x][y] = 1; //place a mine at location
        mines++; //increases mine count
    }while(mines<10); //keep going until we have 10 mines

    for(x=0;x<10;x++){
        for(y=0;y<10;y++){
            cout<<grid[x][y]<<" "; //print out grid
        }
        cout<<endl; //After print a row, do a new line, and start a new row
    }
    cout<<"There are "<<mines<<" Mines\n\n";

    //----- This is where I need to begin the user input
```

```

do{
    cout<<"Input (x, y) entered on separate lines"<<endl; //using variables i for x and j for y, just to
make sure nothing gets messed up
    cin>>i;
    cin>>j;
    i = i-1;
    j = j-1;
    cout<<"i is "<<i<<" and j is "<<j<<endl;
    if(grid[i][j] != 1){
        cout<<"mark is "<<mark<<endl;
        //Test A
        if(grid[i-1][j+1] == 1){
            mark += 1;
            cout<<"mark is "<<mark<<endl;
        }
        //Test B
        if(grid[i][j+1] == 1){
            mark += 1;
            cout<<"mark is "<<mark<<endl;
        }
        //Test C
        if(grid[i+1][j+1] == 1){
            mark += 1;
            cout<<"mark is "<<mark<<endl;
        }
        //Test D
        if(grid[i-1][j] == 1){
            mark += 1;
            cout<<"mark is "<<mark<<endl;
        }
        //Test E
        if(grid[i][j] == 1){
            mark += 1;
            cout<<"mark is "<<mark<<endl;
        }
        //Test F
        if(grid[i-1][j-1] == 1){
            mark += 1;
            cout<<"mark is "<<mark<<endl;
        }
        //Test G
        if(grid[i][j-1] == 1){
            mark += 1;
            cout<<"mark is "<<mark<<endl;
        }
        //Test H
        if(grid[i+1][j-1] == 1){
            mark += 1;
            cout<<"mark is "<<mark<<endl;
        }

        cout<<"mark is "<<mark<<endl;
    }
}

```

```

grid[i][j] = mark;
for(x=0;x<10;x++){
    for(y=0;y<10;y++){
        cout<<grid[x][y]<<" "; //print out grid
    }
    cout<<endl; //After print a row, do a new line, and start a new row
}

```

```

    repeat = 1;
}
else{
    cout<<"You hit a mine, GG"<<endl;
    repeat = 0;
    break;
}
}while(repeat = 1);

```

```

}

```

Trisomy 21 - April 23, 2014, 12:30 pm

For your first question: you could opt to have 2 grids, one with filled with all of the correct numbers/mines, and one filled with 'X's or some other character. When the user chooses a coordinate on the grid with the 'X's, that coordinate location is compared with the real grid and the 'X' is overwritten with the correct data.

shaggy756 - April 23, 2014, 12:36 pm

That's an interesting way to look at it. I'll try that out and see what I can make of it.

Trisomy 21 - April 23, 2014, 12:44 pm

(Apr 23, 2014 12:36 PM)shaggy756 Wrote: [\[CITE\]](#) That's an interesting way to look at it. I'll try that out and see what I can make of it.

You could also use a 2d array of objects as your grid. And each object will hold variables specifying its number(if it is not a mine), whether it is a mine or not, and a boolean value that switches once it is uncovered by the user. This might be cleaner than using two separate arrays but im not sure.

And obviously you can use Boost.Random or the old srand() to randomize the values in each object(excluding the boolean switch).

shaggy756 - April 23, 2014, 12:50 pm

Ahh I like that idea a lot more. How would I go about doing that? Would I put the array inside of the class and use a for loop of some sort?

Trisomy 21 - April 23, 2014, 12:53 pm

(Apr 23, 2014 12:50 PM)shaggy756 Wrote: [\[CITE\]](#) Ahh I like that idea a lot more. How would I go about doing that? Would I put the array inside of the class and use a for loop of some sort?

No, google "array of objects". You will literally be creating an array of instances of your class.

shaggy756 - April 23, 2014, 12:59 pm

Ok awesome! I'll do that now. Thanks for your help!

0xDEAD10CC - April 23, 2014, 6:10 pm

C++11 has the `<random>` header btw... And why would you use arrays here? Have you looked into the STL? There's some valuable containers that could be useful here. Everything is in `main()` too... Why?

Z5T1 - April 25, 2014, 10:16 am

You could just use a third value for undiscovered mines, eg:

0 = no mine

1 = discovered mine

2 = undiscovered mine

As for the second issue, you may find managing the board much easier if you use a console UI library like `ncurses` instead of repeatedly printing the entire gameboard all over. You can find a really great guide on `ncurses` [here](#).

0xDEAD10CC - April 27, 2014, 12:41 am

OP: Are you paying attention to what you're writing? Beginner mistake:

Code ... `}while(repeat = 1);`