Robots

In this assignment you have to convert the previous grid and robot into C++ classes. Most of the functions are the same, but some syntax will change.

Grid:

```
#ifndef CS170_GRID
#define CS170_GRID
namespace CS170 {
  class Grid {
    public:
      Grid(int x, int y);
      ~Grid();
      bool Inside(int x, int y);
      void Mark(int x, int y);
      bool Marked(int x, int y);
    private:
      int x_max, y_max;
      int* data;
  };
}
#endif
```

Create and destroy function are now Grid constructor and destructor. Constructor accepts width and height of the grid. The 3 other functions are the same as before, but there is no need to pass a pointer to the grid anymore, since it's available automatically.

Robots:

```
#ifndef CS170_ROBOT
#define CS170_ROBOT
#include "grid.h"

namespace CS170 {
  class Robot {
    public:
      enum Orientation {NORTH,EAST,SOUTH,WEST};
      Robot(Grid& _grid,int _x=0, int _y=0, Orientation orientation=SOUTH);
      ~Robot();
      void Move(char cmd);
      int GetX();
      int GetY();
      Orientation GetO();
      bool GetStatus();
    private:
      Grid& grid;
      int x,y;
      Orientation heading;
  };
}
#endif
```

- Robot(Grid& _grid,int _x=0, int _y=0, Orientation orientation=SOUTH); constructor. Accepts grid by reference. Since several robots will be using the same world (grid), the grid is created beforehand and then each robot is given access to it by reference,
- ~Robot() - destructor, may be empty
- void Move(char cmd) – as in the previous assignment
- int GetX(), int GetY(), Orientation GetO(), bool GetStatus() - getter methods (instead of QueryPosition). Return x coordinate, y coordinate, orientation, and whether robot is on the grid.