Problem 1: Random Walk

Please submit `randomwalk.cpp` to print the random walk trajectory of a robot, described as follows:

A robot is initially located at position $(0, 0)$ in a grid $[-5, 5] \times [-5, 5]$. The robot can move randomly in any of the directions: up, down, left, right. The robot can only move one step at a time.

For each move of the robot, print the direction of the move and its position after the move. Use formatted output to print the direction (Down, Up, Left or Right) in the left. The direction takes 10 characters in total and fill in the field with empty spaces. The statement to print results in such format is given below:

```
cout << setw(10) << left << "Down" << [robot position] << endl;
cout << setw(10) << left << "Up" << [robot position] << endl;
```

Here "left" is the keyword in C++ for left-aligned output. You need to replace `[robot position]` with the robot position.

If the robot moves back to the original place (0,0), print "Back to the origin!" and stop the program. If it reaches the boundary of the grid, print "Hit the boundary!" and stop the program. A successful run of your code may look like:

```
Right     (1,0)
Up        (1,1)
Right     (2,1)
Down      (2,0)
Up        (2,1)
Up        (2,2)
Down      (2,1)
Left      (1,1)
Left      (0,1)
Left      (-1,1)
Right     (0,1)
Right     (1,1)
Down      (1,0)
Left      (0,0)
Back to the origin!
```

or

```
Left      (-1,0)
Left      (-2,0)
Right     (-1,0)
Left      (-2,0)
Up        (-2,1)
Up        (-2,2)
Down      (-2,1)
Down      (-2,0)
Left      (-3,0)
Right     (-2,0)
Down      (-2,-1)
Right     (-1,-1)
Down      (-1,-2)
Left      (-2,-2)
Down      (-2,-3)
Down      (-2,-4)
Down      (-2,-5)
Hit the boundary!
```

Due to randomness, your results may have a different trajectory path than examples. Your code should generate different robot trajectories for each run.

Hint: Use two int variables x and y to represent robot's position and use the random integer generator to pick a moving direction at each step. Use a loop to keep moving the robot as long as the robot doesn't hit the boundary or go back to the origin.

Problem 2: Factoring of Integers

Please submit `factor.cpp` to ask the user for a positive integer (greater than 1) and then print out all its prime factors.

Please have the output formatted like the following examples: (the `150`, `7` and `27` are user input)

```
Input a positive integer greater than 1: 150
Prime factors: 2 3 5 5
```

```
Input a positive integer greater than 1: 7
Prime factors: 7
```

```
Input a positive integer greater than 1: 27
Prime factors: 3 3 3
```

- **Headers:** You may only use the following headers: `<iostream>`.
- **No invalid inputs:** You do NOT have to worry about the user giving invalid inputs like strings, doubles or integers no larger than 1.
- (Optional, no points) Try to reduce the iterations in your program.

Instructions:

- All code must be written originally by yourself. You are not allowed to (even partially) copy code from anyone else. Incident of cheating or plagiarism will be reported to the Dean's office and results in a zero grade in this assignment.
- (5pt) Write two programs to solve above questions. Name your files `randomwalk.cpp` and `factor.cpp`, and submit them to Gradescope. You must name the files EXACTLY as instructed, otherwise 2.5 points will be deducted each.
- (5pt) Add declaration in the beginning of each cpp file to show the ownership.
- (10pt) Write your code with good practice as introduced in class.
- (Problem 1 40pt & Problem 2 40pt) Code compiles with Visual Studio 2022 and solves the questions. Students may lose the majority of points if their code doesn't compile with VS 2022.

  To receive full credits, the output must look EXACTLY the same as instructed above, including words, spaces, symbols, etc. Your code should not only work for the above examples, but also work for other different inputs.