Sample: C++ - Exception Handling

Task:

These questions are based upon Exception Handling in c++

Question:
One argument used to justify the fact that the stack data structure in the standard library does not throw exceptions is that it is easy to add these facilities. Create a class SafeStack that implements a stack of strings. Use an instance of stack<string> to hold the underlying values, and implement the same interface as that data type. However, your class should throw an exception if an attempt is made to remove a value from an empty stack.

Question 2:
Write a program that can be used to demonstrate that the order in which catch clauses are listed is important.

Solution:

Question 1:

FILE SafeStack.h

```
#pragma once
#include <stack>
#include <string>
#include "EmptyStackException.h"
using namespace std;

class SafeStack
{
public:

	bool empty();
	void emplace(string &&val);
	void pop();
	void push(string val);
	int size();
	void swap(SafeStack ss);
	string top();

private:
	stack<string> s;
};
```

FILE SafeStack.cpp

```
#include "SafeStack.h"

bool SafeStack::empty(){
	return s.empty();
}

void SafeStack::emplace(string &&val){
	s.emplace(val);
}

void SafeStack::pop(){
	if(s.empty())
```

```
			throw EmptyStackException();
		s.pop();
}

void SafeStack::push(string val){
		s.push(val);
}

int SafeStack::size(){
		return s.size();
}

void SafeStack::swap(SafeStack ss){
		s.swap(ss.s);
}

string SafeStack::top(){
		return s.top();
}
```

File EmptyStackException.h

```
#pragma once
#include <exception>
using namespace std;

class EmptyStackException : public exception
{
public:
		const char * what () const throw ()
		{
			return "Stack is empty";
		}
};
```

Question 2:

FILE Driver.cpp

```
#include "SafeStack.h";
#include "EmptyStackException.h"
#include <iostream>;

int main(){
		SafeStack ss;
		for(int i= 0; i < 2; i++){
			ss.push(""+i);
		}
		for(int i = 0; i < 3; i++){
			cout<<"Stack size = "<<ss.size()<<endl;
			try{
				ss.pop();
			}catch(EmptyStackException e){
				cout<<e.what()<<endl;
			}
		}
		return 0;
}
```

Result: