Please read over the code to the right, and assume that this is compiled to **a.out**.

Question 1: What is the output of:

```
UNIX> echo A | ./a.out
```

Question 2: What is the output of:

```
UNIX> echo B | ./a.out
```

Question 3: What is the output of:

```
UNIX> echo C | ./a.out
```

Question 4: What is the output of:

```
UNIX> echo D | ./a.out
```

```
#include <iostream>
using namespace std;

string a(const string &s)
{
  if (s == "A") throw (string) "B";
  try {
    if (s == "C") throw (string) "D";
    return "E";
  } catch (const string &t) {
    cout << t;
  }
  return "F";
}

int main()
{
  string t;

  cin >> t;
  try {
    cout << a(t);
  } catch (const string &s) {
    cout << s;
  }
  cout << endl;
  return 0;
}
```

Answers

```
UNIX> g++ code.cpp
UNIX> echo A | ./a.out        # Question 1
B
UNIX> echo B | ./a.out        # Question 2
E
UNIX> echo C | ./a.out        # Question 3
DF
UNIX> echo D | ./a.out        # Question 4
E
UNIX>
```

Explanations:

- When "A" is entered and passed to the procedure **a()**, the procedure throws the string "B". That **throw** statement is not inside a **try**, so the exception is passed to the calling procedure **main()**. **main()** catches the exception and prints "B". It then prints a newline and exits.
- When "B" is entered and passed to the procedure **a()**, both **if** statements are false, so no exceptions are thrown. **a()** returns "E" to the caller, which prints it out, and then prints a newline.
- When "C" is entered and passed to the procedure **a()**, the procedure throws the string "D". That is caught in **a()**, which prints "D". After the **catch** code, it returns "F". **main()** prints "R" and a newline.
- Any string that is not "A" or "C" will be identical to question 2.