- Exercise

Let's solve a template specialization problem in this lesson.

WE'LL COVER THE FOLLOWING

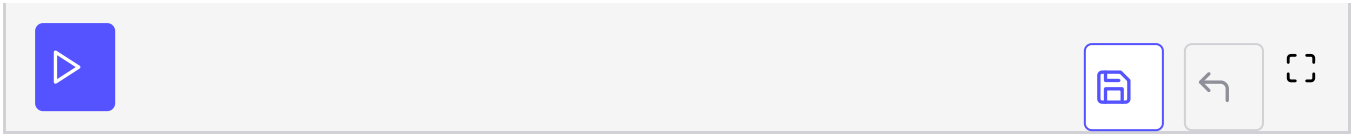
Problem Statement

The class template `Type` in the code below returns to each type the name `unknown`.

- Use the class template `Type` as a starting point to write a type introspection system with the help of partial and full specialization.
- You need to write code for `int`, `double`, an arbitrary class named `Account`, `pointer`, `const`, and `string`.

```
#include <iostream>
#include <string>

// Implement with full and partial specialization
// Write your code here

template<typename T>
struct Type{
  std::string getName() const {
    return "unknown";
  }
};

int main(){

  std::cout << std::boolalpha << std::endl;

  // call objects for each defined templetes here

  // An example of float is given below which returns "unknown"
  Type<float> tFloat;
  std::cout << "tFloat.getName(): " << tFloat.getName() << std::endl;

  std::cout << std::endl;
```

```
}
```



In the next lesson, we'll look at the solution to this exercise.