Review Test Submission: Midterm Sample #2: Custom Types

User	Yihsun Lee
Course	Object-Oriented Software Development Using C++
Test	Midterm Sample #2: Custom Types
Started	6/17/21 10:07 AM
Submitted	6/21/21 2:56 PM
Due Date	6/23/21 11:59 PM
Status	Needs Grading
Attempt Score	Grade not available.
Time Elapsed	**100 hours, 49 minutes out of 1 hour** OVER TIME
Instructions	You are allowed to use any tool and material available to you during the test. The following are considered plagiarism: • exchanging messages in any form with another person during the test. • allowing access to your questions/solutions to somebody else before the test due date. • acquiring the test questions/solutions that somebody else had before the test due date.
Results Displayed	Submitted Answers

Question 1

Needs Grading

Inspect the following type definition below which is used to store an on/off switch.

```
enum class StateType
{
    boolean,
    character,
    number
};

typedef struct
{
```

```
    char name[20];
    StateType sType;
    union
    {
        bool stateAsBool;
        char stateAsChar;
        int stateAsNum;
    } sState;
} Switch;
```

A switch is considered **ON** if the value satisfies one of the conditions below.

State Type	sState ("on" value)
boolean	true
character	'O'
number	1

Use the `Switch` definition above to build a class that models a home stereo system. Such a system will have output channels such as TV, DVD, Bluetooth device, Computer, etc. Each `Switch` object represents the current state of an output channel.

Basic Details

Your `HomeStereo` class has the following data members

- an array of **dynamically allocated** `Switch`-typed values
- a **non-negative integer** that stores the number of values in the array

A `HomeStereo` object can be created using a **2-argument Constructor**.

Public Member Functions

- `displayOutputState()`: receives an `ostream` **reference** (defaults to standard output) and returns an `ostream` **reference**. This function inserts the state of all switches into the given output stream. The output should be in the following form.

```
Channel Name: xxxxxx - State [on/off]< endl >
Channel Name: xxxxxx - State [on/off]< endl >
...
```

Other Features

Include in your design all special member functions required to manage your objects.

Misc

You are allowed to add as many *private* members as your design requires!

Put in the answer box the content of your header file and implementation file. Both files must be properly created according to C++ standard and best practices.

Selected Answer:

```
#ifndef SDDS_HomeStereo_h
#define SDDS_HomeStereo_h

#include <iostream>

namespace sdds{
enum class StateType
{
   boolean,
   character,
   number
};

typedef struct
{
   char name[20];
   StateType sType;
   union
   {
      bool stateAsBool;
      char stateAsChar;
      int stateAsNum;
   } sState;
} Switch;

class HomeStereo{
   Switch* m_switch{};
   size_t numArr{0};
public:
   HomeStereo(){};
   HomeStereo(const Switch*, size_t);
   // copy constructor
   HomeStereo(const HomeStereo&);
   // copy assignment operator
   HomeStereo& operator=(const HomeStereo&);
   // move constructor
   HomeStereo(HomeStereo&&);
   // move assignment operator
   HomeStereo& operator=(HomeStereo&&);
```

```
   HomeStereo& operator=(HomeStereo&&);
   ~HomeStereo();
   std::ostream& displayOutputState(std::ostream&)const;
};
}

#endif /* SDDS_HomeStereo_h */



#include <iostream>
#include <cstring>
#include "HomeStereo.h"

using namespace std;
namespace sdds{

HomeStereo::HomeStereo(const Switch* obj, size_t num){
   numArr = num;
   m_switch = new Switch[num];
   for(size_t i = 0; i < num; i++){
      m_switch[i].sType = obj[i].sType;
      m_switch[i].sState = obj[i].sState;
      strncpy(m_switch[i].name, obj[i].name, 20);
   }
}

HomeStereo::HomeStereo(const HomeStereo& src){
   *this = src;
}

HomeStereo& HomeStereo::operator=(const HomeStereo&
src){
   if(this != &src){
      numArr = src.numArr;
      delete[] m_switch;
      m_switch = new Switch[numArr];
      for(size_t i = 0; i < numArr; i++){
         m_switch[i] = src.m_switch[i];
      }
   }
   return *this;
}

HomeStereo::HomeStereo(HomeStereo&& src){
   *this = std::move(src);
}

HomeStereo& HomeStereo::operator=(HomeStereo&& src){
   if(this != &src){
      delete[] m_switch;
```

```
        delete[] m_switch;
        m_switch = src.m_switch;
        src.m_switch = nullptr;
        numArr = src.numArr;
        src.numArr = 0;
    }
    return *this;
}

HomeStereo::~HomeStereo(){
    delete[] m_switch;
    m_switch = nullptr;
}

std::ostream& HomeStereo::displayOutputState(std::ostream&
os)const{
    for(size_t i = 0; i < numArr; i++){
        os << "Channel Name: " << m_switch[i].name << " - State [
";
        if(m_switch[i].sState.stateAsBool){
            os << "on ]\n";
        } else if(m_switch[i].sState.stateAsChar == 'O'){
            os << "on ]\n";
        } else if(m_switch[i].sState.stateAsNum == 1){
            os << "on ]\n";
        } else {
            os << "off ]\n";
        }
    }
    return os;
}
}
```

← OK