**Отчет по лабораторной работе №3**

**“ Конструирование класса, моделирующего работу устройства”**

1. **Постановка задачи:**

Разработать модель работы устройства, состоящую из 3­4 свойства, наиболее существенных для описания поведения устройства, и 3­6 методов, моделирующих поведение объекта.

Определить набор допустимых значений для каждого свойства.

На основе модели сконструировать класс. Доступ к свойствам класса должен быть закрытым (private), к методам – открытым (public); изменение значений свойств (состояния) объекта осуществляется соответствующими методами.

Разработать консольное приложение, демонстрирующее работу объекта, в составе: главный модуль, содержащий функцию main,

* + модуль, содержащий разработанный класс.

Программа должна выполнять следующие действия:

* + создание объекта;
  + демонстрация текущего состояния объекта с помощью сообщений на экране;
  + действия над объектом, меняющие его состояние.

Выбор действия над объектом осуществить через меню в цикле, чтобы предоставить пользователю возможность выбирать любую последовательность действий.

10 вариант: Кондиционер

1. **Структуры, использованные в программе:**

Класс рекомендуется помещать в отдельный модуль, состоящий из двух частей: интерфейса модуля и реализации модуля, имеющих одно имя, но разные расширения. В С++ интерфейсная часть модуля класса помещается в заголовочный файл (\*.h), содержит объявление класса, директивы препроцессора.

## Диаграмма компонентов языка UML

UML (Unified Modelling Language) — это специальный графический язык, предназначенный для моделирования объектно­ориентированных программ и систем. Главные элементы UML — диаграммы, представляющие программу или систему с разных точек зрения. Существуют статические и динамические диаграммы. Статическая диаграмма представляет программу вне зависимости от времени. Динамические диаграммы описывают поведение системы во времени.

Диаграмма компонентов — один из видов диаграмм UML. Это статическая диаграмма, демонстрирующая файловый состав проекта (файлы с исходным кодом, двоичные файлы, динамически загружаемые библиотеки, файлы с данными, и.т.д. и их связи). Для обозначения компонента на диаграмме используется прямоугольник с именем файла или компонента внутри. Для обозначения связей компонентов используются различные типы линий со стрелками и без.

Один из видов связи — «использование», отражает зависимость одного компонента от другого. Например, при сборке проекта из исходного кода эта зависимость проявляется при подключении модуля класса к главному модулю (директива #include). Связь типа использование обозначается пунктирной линией со стрелкой в направлении от зависимого файла к независимому

1. **Описание программы**

Программа разработана на языке C++ в среде Microsoft Visual Studio 19 для OC Windows, в ней реализованы классы public и private. С помощью методов, используемых в классе public, осуществляется изменение состояния прибора. В программе использованы следующие методы:

- Изменение температуры(ChangeTemperature)

- Изменение скорости потока воздуха(SetAirSpeed)

- Включение/Выключение увлажнения(OnOffHumidifier)

Для придания реалистичности, изменение температуры воздуха проходит с задержкой. Такой эффект добивается с помощью Sleep, которой подаётся модуль разницы температур.

Программа использует цикл while. Взаимодействие с пользователем осуществляется с помощью switch case, а для вывода используется cout вместе с одним из методов получения необходимой переменной

1. **Результаты работы программы**

На рисунках ниже представлена информация по вызову различных функций программы:

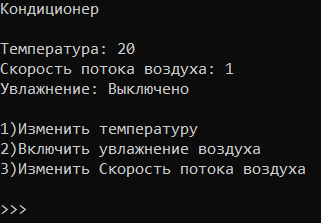


Рис.1 Начало работы с программой

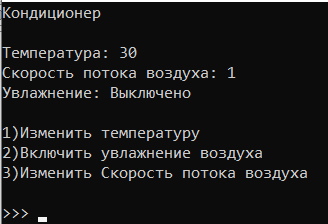


Рис.2 Пример работы функции “Изменение температуры”

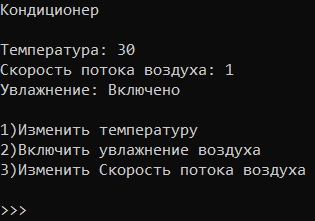


Рис. 3 Пример работы функции “Увлажнение воздуха”

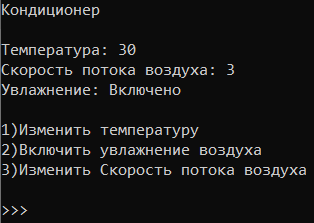


Рис. 4 Пример работы функции “Изменение скорости потока воздуха”

1. **Выводы**В ходе проделанной работы были получены следующие выводы:
2. Изучены методы работы со структурами данных
3. Освоена среда Visual Studio
4. Разработана программа, симулирующая работу кондиционера, с возможностью редактирования следующих функций:

- Изменение температуры (ChangeTemperature)

- Изменение скорости потока воздуха (SetAirSpeed)

- Включение/Выключение увлажнения (OnOffHumidifier)

Для придания реалистичности, изменение температуры воздуха проходит с задержкой. Такой эффект добивается с помощью Sleep, которой подаётся модуль разницы температур.

1. **Код программы**

#include <iostream>

#include <windows.h>

#include <string>

class Сonditioner

{

public:

void OnOffHumidifier()

{

if (onHumidifier)

{

onHumidifier = 0;

}

else

{

onHumidifier = 1;

}

}

void ChangeTemperature(float newTemperature)

{

Sleep(1000 \* abs(Temperature - newTemperature));

Temperature = newTemperature;

}

bool GetonHumidifier()

{

return onHumidifier;

}

float GetTemperature()

{

return Temperature;

}

void SetAirSpeed(int newAirSpeed)

{

AirSpeed = newAirSpeed;

}

int GetAirSpeed()

{

return AirSpeed;

}

private:

int AirSpeed = 1;

float Temperature = 20.0;

bool onHumidifier = 0;//Увлажнение воздуха(есть в некоторых кондиционерах)

};

int main()

{

setlocale(LC\_ALL, "rus");

Сonditioner Cond1;

int choice;

while (1)

{

std::cout << "Кондиционер\n\n";

std::cout << "Температура: " << Cond1.GetTemperature() << "\n" << "Скорость потока воздуха: " << Cond1.GetAirSpeed() << "\nУвлажнение: ";

if (Cond1.GetonHumidifier())

std::cout << "Включено\n\n";

else

std::cout << "Выключено\n\n";

std::cout << "1)Изменить температуру\n2)Включить увлажнение воздуха\n3)Изменить Скорость потока воздуха\n\n>>> ";

std::cin >> choice;

switch (choice)

{

case 1:

{

float newTemperature;

std::cout << "Введите новую температуру: ";

std::cin >> newTemperature;

Cond1.ChangeTemperature(newTemperature);

system("cls");

break;

}

case 2:

{

Cond1.OnOffHumidifier();

system("cls");

break;

}

case 3:

{

int newAirSpeed;

std::cout << "Введите скорость(1-5): ";

std::cin >> newAirSpeed;

if (newAirSpeed < 6 && newAirSpeed > 0)

{

Cond1.SetAirSpeed(newAirSpeed);

system("cls");

}

else

{

system("cls");

std::cout << "Ошибка ввода\n\n";

}

break;

}

default:

{

system("cls");

std::cout << "Ошибка ввода\n\n";

break;

}

}

}

}