МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Российский государственный университет им. А.Н. Косыгина   
(Технологии. Дизайн. Искусство)»**

Институт ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ЦИФРОВЫЕ ТРАНСФОРМАЦИИ

**Отчет по лабораторной работе № 6**

**по дисциплине «Объектно-ориентированное программирование»**

**Тема: «Классы. Приведение типов. Реализация класса Stroka»**

Выполнил: Сидоров Д. С., группа ИТС-122

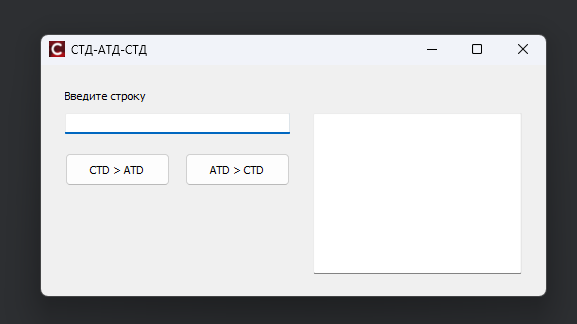
Проверил: к.т.н., доц. Семёнов А. А.

Москва, 2025г

**Классы. Приведение типов. Реализация класса Stroka**

Изучить и повторить проиллюстрированный в видеозаписи пример, демонстрирующий реализацию следующей задачи.

На основе типа char создайте класс Stroka. Перегрузите операцию приведения строки типа char к типу Stroka и наоборот. Напишите визуальное приложение (Windows VCL Application) для проверки этого класса. Разбить по модулям (unit1 и unit2).

  
Рисунок 1. Начальное окно программы

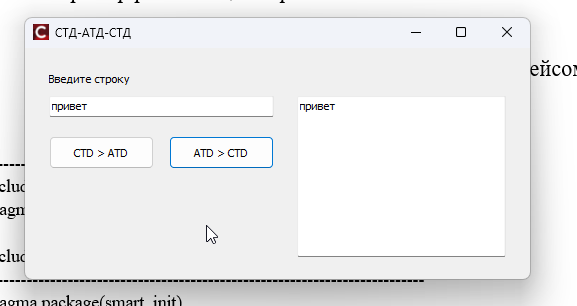


Рисунок 2. Приложение с активным интерфейсом

**Листинг кода**

**Unit1.cpp**

///---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

Caption = "СТД-АТД-СТД";

Label1->Caption = "Введите строку";

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::CTDtoATDButtonClick(TObject \*Sender)

{

AnsiString inputStr = InputEdit->Text;

const char\* stdString = inputStr.c\_str();

Stroka strokaObj = stdString;

OutputMemo->Clear();

OutputMemo->Lines->Add(strokaObj.GetStr());

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::ATDtoCTDButtonClick(TObject \*Sender)

{

AnsiString inputStr = InputEdit->Text;

Stroka strokaObj(inputStr.c\_str());

const char\* stdString = strokaObj;

OutputMemo->Clear();

OutputMemo->Lines->Add(stdString);

}

//---------------------------------------------------------------------------

**Unit1.h**

//---------------------------------------------------------------------------

#ifndef Unit1H

#define Unit1H

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include "Unit2.h"

//---------------------------------------------------------------------------

class TForm1 : public TForm

{

\_\_published: // IDE-managed Components

TEdit \*InputEdit;

TButton \*CTDtoATDButton;

TButton \*ATDtoCTDButton;

TMemo \*OutputMemo;

TLabel \*Label1;

void \_\_fastcall CTDtoATDButtonClick(TObject \*Sender);

void \_\_fastcall ATDtoCTDButtonClick(TObject \*Sender);

private: // User declarations

public: // User declarations

\_\_fastcall TForm1(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm1 \*Form1;

//---------------------------------------------------------------------------

#endif

**Unit2.cpp**

//---------------------------------------------------------------------------

#pragma hdrstop

#include "Unit2.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

// Конструктор по умолчанию

Stroka::Stroka() : data(nullptr), length(0)

{

data = new char[1];

data[0] = '\0';

}

// Конструктор с параметром

Stroka::Stroka(const char\* str) : data(nullptr), length(0)

{

if (str) {

length = strlen(str);

data = new char[length + 1];

strcpy(data, str);

} else {

data = new char[1];

data[0] = '\0';

}

}

// Конструктор копирования

Stroka::Stroka(const Stroka& other) : data(nullptr), length(other.length)

{

if (other.data) {

data = new char[length + 1];

strcpy(data, other.data);

} else {

data = new char[1];

data[0] = '\0';

}

}

// Деструктор

Stroka::~Stroka()

{

delete[] data;

}

// Оператор присваивания

Stroka& Stroka::operator=(const Stroka& other)

{

if (this != &other) {

delete[] data;

length = other.length;

if (other.data) {

data = new char[length + 1];

strcpy(data, other.data);

} else {

data = new char[1];

data[0] = '\0';

}

}

return \*this;

}

// Оператор преобразования из char\* в Stroka

Stroka& Stroka::operator=(const char\* str)

{

delete[] data;

if (str) {

length = strlen(str);

data = new char[length + 1];

strcpy(data, str);

} else {

length = 0;

data = new char[1];

data[0] = '\0';

}

return \*this;

}

// Оператор преобразования из Stroka в char\*

Stroka::operator const char\*() const

{

return data;

}

// Метод для получения строки

const char\* Stroka::GetStr() const

{

return data;

}

// Метод для получения длины

size\_t Stroka::GetLength() const

{

return length;

}

**Unit2.h**

//---------------------------------------------------------------------------

#ifndef Unit2H

#define Unit2H

#include <string.h>

#include <vcl.h>

//---------------------------------------------------------------------------

// Класс Stroka, основанный на типе char

class Stroka

{

private:

char\* data; // Указатель на строку символов

size\_t length; // Длина строки

public:

// Конструктор по умолчанию

Stroka();

// Конструктор с параметром

Stroka(const char\* str);

// Конструктор копирования

Stroka(const Stroka& other);

// Деструктор

~Stroka();

// Оператор присваивания

Stroka& operator=(const Stroka& other);

// Оператор преобразования из char\* в Stroka

Stroka& operator=(const char\* str);

// Оператор преобразования из Stroka в char\*

operator const char\*() const;

// Метод для получения строки

const char\* GetStr() const;

// Метод для получения длины

size\_t GetLength() const;

};

#endif

**Вывод:** В ходе выполнения лабораторной работы был создан класс Stroka на основе типа char, демонстрирующий механизм перегрузки операций в C++. Были реализованы операторы преобразования для конвертации между стандартным типом char\* и пользовательским типом Stroka с применением перегрузки операторов приведения типа. Разработанное VCL-приложение позволяет наглядно продемонстрировать работу созданного класса через графический интерфейс, предоставляя возможность ввода строки и проверки операций преобразования в обоих направлениях. Проект структурирован в соответствии с принципами модульного программирования и разделен на два модуля: Unit1 содержит код формы и управление интерфейсом, а Unit2 включает реализацию класса Stroka с необходимыми методами и операторами. Данная работа показывает, как можно расширять возможности языка C++ путем создания пользовательских типов данных с собственными операциями преобразования.