ворка React [2]. Для создания компонентов при разработке интерфейса используется компонентная библиотека Material-UI. Для улучшения маршрутизации в клиентской части проекта используется библиотека Reactrouter [4].

Веб-приложение ориентировано на учащихся и сотрудников IT-школы для повышения качества проверки знаний.

ЛИТЕРАТУРА

1.WhatSpringcando [Электронный ресурс]. – Режим доступа: https://spring.io/. Дата доступа: 22.03.2021.

2.React [Электронный ресурс]. – Режим доступа: https://reactjs.org/. Дата доступа: 22.03.2021.

3. ReactRouter [Электронный ресурс]. – Режим доступа: https://reactrouter.com/. Дата доступа: 01.04.2021.

УДК 004.588

Студ. Е.А. Чернявский
Науч. рук. ст. преп. А.С. Наркевич
(кафедра программной инженерии, БГТУ)

РЕАЛИЗАЦИЯ КОМПИЛЯТОРА С ЯЗЫКА ПРОГРАММИРОВАНИЯ CEA-2020

Разработка компилятора началась с написания спецификации языка программирования CEA-2020. Язык программирования реализован процедурным, универсальным, строго типизированным, компилируемым на язык ассемблера и не объектно-ориентированным. Используемая кодировка – ASCII. В наличии имеет два типа данных, строковый string максимальным размером 255 бит, и целочисленный integer, в диапазоне от 0 до 255 (размер 1 байт). Язык должен поддерживать арифметические и побитовые операции, также функции, цикл и условный оператор. Должна присутствовать стандартная библиотека с функциями возведения числа в n-ую степень, взятия корня n-ой степени, перевода числа в строку, вывод чисел и строк в консольное окно. Кроме этого, должна быть реализована конкатенация и сравнение строк.

Первая стадия работы компилятора – это лексический анализ. Для этого был реализован лексический анализатор, на вход которого подается исходный код программы. Анализатор должен пропускать лишние символы, а также выдавать ошибку при наличии запрещенных символов в коде, расставлять пробелы между символами сепараторами и собрать информацию об исходном и преобразованном коде в виде таблиц лексем и идентификаторов. Данные таблицы заполняют-

ся, в моем случае, при помощи флажков. При встрече определенных лексем выставляются определенные флажки, и убираются соответственно. Таким образом, можно отследить идентификаторы. Вызов функций заменяется на специальный символ ‘@’ и хранит в себе идентификаторы всех переданных параметров в себе, чтобы впоследствии можно было легко обратиться к ним.

Вторая стадия – синтаксический анализ. Задача синтаксического анализатора заключается в проверке исходного кода на соответствие правилам грамматики. Входной информацией являются таблицы лексем и идентификаторов. Выходная информация анализатора – дерево разбора. В моем случае синтаксический анализатор реализован при помощи конечного магазинного автомата с памятью. При обработке кода, в стек закладывается правило, и далее анализатор проходится по ленте таблицы лексем, сравнивая лексемы с лентой правила. При совпадении происходит сохранение состояния, иначе, если символ из ленты таблицы лексем не совпадает с символом ленты правила, анализатор возвращается к предыдущему сохраненному состоянию и пытается подобрать подходящее правило и продолжить свою работу. Если при переборе правил не нашлось подходящего, то компиляция завершается с ошибкой.

Третья стадия компиляции – семантический анализ. После успешного построения дерева разбора начинается проверка логики выражений. Тип возвращаемых значений функций должен совпадать с типом функции, присваиваемое значение должно соответствовать с типом идентификатора. Алгоритм семантического анализатора похож на лексический анализатор. Он также проходится по лексемам, расставляя флажки и делая выводы. При неправильных конструкциях, компилятор завершает свою работу с ошибкой.

Последняя стадия – генерация кода. Для того, чтобы мой язык имел возможность вычислять сложные конструкций, все выражения преобразуются к обратной Польской записи. Данная запись избавляет выражение от скобок и расставляет операции выражения в зависимости от расставленных приоритетов. Преобразование к обратной Польской записи выполняется за один проход по таблице лексем, заменяя ленту таблицы на преобразованную, а также освобождает память, если лента символов уменьшается на несколько символов. Генератор кода – часть транслятора, выполняющая генерацию ассемблерного кода на основе полученных данных на предыдущих этапах трансляции. На вход генератора подаются таблица лексем и таблица идентификаторов. В соответствии с таблицей лексем строится выходной файл на языке ассемблера, который будет являться результатом работы транс-

лятора. В случае возникновения ошибок генерация кода не будет осуществляться.

Для удобства работы с компилятором CEA-2020 был разработан интерфейс на языке программирования C#. С его помощью можно быстро сгенерировать код.

ЛИТЕРАТУРА

1. Let’sBuild A SimpleInterpreter [Электронный ресурс] - Режим доступа: https://ruslanspivak.com/lsbasi-part1/ (дата обращения 22.10.2020).

2. В.А.Серебряков, М.П.Галочкин Основы конструирования компиляторов [Электронный ресурс] - Режим доступа: http://citforum.ru/programming/theory/serebryakov/ (дата обращения 14.11.2020)

УДК 004.4

Студ. Д.В. Питаленко
Науч. рук. ст. преп. И.Г. Сухорукова
(кафедра программной инженерии, БГТУ)

ВЕБ-ПРИЛОЖЕНИЕ «DANCE EVENTS»

Танец сопровождал человека всегда. В разные эпохи он являлся частью культуры, религии, воспитания, становился профессией, терапией, развлечением, спортом, искусством. Откуда в нас живет потребность к движению? Что мы ищем в танце и почему связываем его с чем-то радостным и счастливым, с эмоциональным подъемом, с праздником?

Танец являлся формой коммуникации до того, как появилась устная речь. Наверное, поэтому язык танца понимают больше людей, чем какой бы то ни было устный язык.

Целью данного проекта является создание системы для просмотра расписания мастер-классов и регистрации по разнообразным танцевальным направлениям.

Серверная часть приложения написана на ASP.NET Core 3.1, язык программирования C# [1]. В качестве СУБД используется MSSQL Server. Для работы с базой данных используется Entity Framework Core [2].

Также в проекте демонстрируется применение n-layer архитектуры. Для разработки клиентской части используется React [3].

В результате выполнения работы было разработано приложение, позволяющее реализовывать основные функции системы для осуществления администрирования событий танцевальных студий.