Содержание:

ВВЕДЕНИЕ

Актуальность темы исследования: Прогресс компьютерных технологий определил процесс появления новых разнообразных знаковых систем для записи алгоритмов – языков программирования. Смысл появления такого языка – оснащенный набор вычислительных формул дополнительной информации, превращает данный набор в алгоритм.

Язык программирования — формальная знаковая система, предназначенная для описания алгоритмов в форме, которая удобна для исполнителя (например, компьютера). Язык программирования определяет набор лексических, синтаксических и семантических правил, используемых при составлении компьютерной программы. Он позволяет программисту точно определить то, на какие события будет реагировать компьютер, как будут храниться и передаваться данные, а также какие именно действия следует выполнять над этими при различных обстоятельствах.

Язык программирования служит двум связанным между собой целям: он дает программисту аппарат для задания действий, которые должны быть выполнены, и формирует концепции, которыми пользуется программист, размышляя о том, что делать. Первой цели идеально отвечает язык, который настолько "близок к машине", что всеми основными машинными аспектами можно легко и просто оперировать достаточно очевидным для программиста образом. Второй цели идеально отвечает язык, который настолько "близок к решаемой задаче", чтобы концепции ее решения можно было выражать прямо и коротко.

Со времени создания первых программируемых машин человечество придумало уже более двух с половиной тысяч языков программирования. Каждый год их число пополняется новыми. Некоторыми языками умеет пользоваться только небольшое число их собственных разработчиков, другие становятся известны миллионам людей. Профессиональные программисты иногда применяют в своей работе более десятка разнообразных языков программирования.

Целью данной работы является изучение языков гипертекстовой разметки. Для достижения поставленной цели, были выделены следующие задачи:

- рассмотреть теоретические аспекты исследования языков программирования;

- провести анализ проектирование информационно-ресурсной среды средствами языков программирования delphi и гипертекстовой разметки HTML.

Объект исследования - языки гипертекстовой разметки.

Предмет исследования - разработка сайтов на основе языка гипертекстовой разметки HTML.

Структура работы состоит из введения, основной части, заключения и списка литературы.

Теоретической и методологической базой данной работы послужили труды российских и зарубежных авторов в области программирования, материалы периодических изданий и сети Интернет.

ГЛАВА 1. ТЕОРЕТИЧЕСКИЕ АСПЕКТЫ ИССЛЕДОВАНИЯ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

1.1 История развития

Ранее говорилось, что основным инструментом общения человека с компьютером является программа. В общем случае под программой понимается некая систематизированная совокупность команд, входящих в список доступных для исполнения на конкретном устройстве. Набор таких команд называется системой команд. Система команд первых компьютеров была очень примитивна, она состояла из набора элементарных арифметических операций и операций пересылки данных.

Первым программистом в истории человечества принято называть Аду Лавлейс, которая составила предписания для аналитической машины Чарльза Бэббиджа. Стоит отметить, что данное вычислительное устройство существовало лишь в проекте. В его основе лежали не электрические сигналы, а механические узлы. Кроме того, машина Бэббиджа должна была быть оснащена паровым двигателем (в 1830-е годы это было вершиной техники). Для ввода программы предполагалось использование перфокарт. Однако данная машина так и не была спроектирована.

Первые реально спроектированные машины программировались при помощи перемычек и тумблеров. Поэтому появилась идея использовать мнемонические обозначения команд, что существенно упростило бы процесс программирования. Так появились языки ассемблера, которые позволили обозначать ячейки памяти компьютера не их адресами, а символическими именами, которые в дальнейшем могли использоваться в программах. Термин ассемблер происходит от английского слова assembler, обозначающего сборку частей в единое целое. Сам процесс сборки называется ассемблированием.

Переход от машинных кодов к ассемблеру позволил увеличить производительность труда программистов и сократить время написания программ. Кроме того, языки ассемблеров позволили увеличить качество и надежность программ за счет меньшего количества возможностей внесения ошибок в программу. Однако было и несколько недостатков. Так, например, программа, написанная на ассемблере, не является понятной компьютеру, и требует переводчика. Также важно отметить, что программы на ассемблере не являлись переносимыми. Это говорит о том, что при смене вычислительной машины программы становились бесполезными, и их приходилось переписывать заново [17].

Следующим этапом стало появление языка Фортран, основным предназначением которого была реализация математических вычислений. Он отличался хорошим качеством получаемых программ, а также развитыми средствами ввода-вывода информации. Кроме того, Фортран содержал собственную библиотеку стандартных программ. Данный язык разрабатывался в 1954-1956 годы крупной группой специалистов компании IBM под руководством Дж. В. Бэкуса. В июне 1956 г. была выпущена вторая версия данного языка – Фортран II, которая отличалась наличием подпрограмм, а также операторов связи между программными единицами [11].

В силу того, что Фортран и языки ассемблера требовали специальной подготовки, появилось новое направление языков, предназначенных непосредственно для обучения программированию. Примерами таких языков являются Pascalи Basic. Важно отметить, что эти языки используются в учебных учреждениях и в настоящее время.

Параллельно с развитием языков программирования велись разработки операционных систем, что послужило появлению системы UNIX. Данная система была написана на языке ассемблера, что служило барьером в ее изучении. Поэтому для упрощения системного программирования Д. Ритчи и Б. Керниган разработали язык С, на котором данная система была переписана. Важно отметить, что UNIX-

системы используются и сегодня.

Дальнейшее развитие языков программирования связано с появлением объектно-ориентированной технологии, которая должна была упростить написание крупных программ промышленных масштабов. Примером одного из первых языков объектно-ориентированного программирования является C++, разработанный Б. Страуструпом.

Последнее десятилетие XX века ознаменовалось развитием глобальной сети Internet, что также послужило толчком к созданию новых технологий. В этот период максимальную популярность обрел язык Java, который позволяет в кратчайшие сроки писать крупные приложения без опасений навредить системе. Данный язык характеризуется переносимостью своих программ [18].

1.2 Классификация языков программирования

Существует несколько признаков классификации языков программирования, рассмотрим лишь основные из них.

По степени абстракции от аппаратуры принято выделять следующие виды языков:

- низкого уровня:
  - машинно-зависимые – представляют собой команды, записанные непосредственно на языке конкретного процессора. Эти команды состоят из нулей и единиц. Фактически, команды таких языков представляют собой свод правил кодирования инструкций для определенного типа ЭВМ при помощи чисел;
  - машинно-ориентированные – команды, записанные на языке близком к процессору. Основное отличие от машинно-зависимых языков заключается в том, что машинные коды в таких языках заменены буквенными обозначениями;
- высокого уровня – машинно-независимые языки, имитирующие естественный человеческий язык. Данные языки используют некоторые слова разговорного языка, а также общепринятые математические символы. Языки высокого уровня ближе к человеку, а не к компьютеру. Они позволяют записывать команды в текстовом режиме, а также предоставляют возможность определять сложные структуры данных;

- сверхвысокого уровня – еще один вид машинно-независимых языков программирования. Команды таких языков исполняются на абстрактных машинах, при этом доступ к памяти полностью скрыт.

Еще одним признаком классификации является парадигма программирования. С этой точки зрения все языки программирования делятся на:

- процедурные – проблемно-ориентированные языки, облегчающие исполнение процедур. Они бывают нескольких видов:
    - структурные – один оператор позволяет записывать целые алгоритмические структуры – циклы, ветвления и т.п.;
    - операционные – используют несколько операторов для записи одной алгоритмической конструкции. Неструктурное программирование позволяет использовать в явном виде оператор безусловного перехода;
    - параллельные – языки реального времени, поддерживающие параллельные вычисления. Программа, написанная на одном из таких языков, представляет собой совокупность описаний процессов, которые могут исполняться одновременно;
- непроцедурные – описывают саму задачу. Делятся на:
    - функциональные – программы на таких языках обычно вычисляют значение некоторой функции, задающейся в виде композиции более простых функций. Одним из ключевых элементов данного языка является наличие рекурсии;
    - логические – языки, базирующиеся на классической логике. Основным их применением является логический вывод, например, экспертные системы. С точки зрения логического подхода программа представляет собой совокупность логических высказываний и правил[3];
- объектно-ориентированные – языки, в основе которых лежит понятие объектов и их свойств, а также отношений между этими объектами [4].

1.3 Алгоритмы языков программирования

В основе любой компьютерной программы лежит алгоритм. Алгоритмом принято называть систему четких однозначных указаний, определяющих последовательной действий над некоторыми объектами, за конечно количество итераций приводящих к желаемому результату [6]. Алгоритмы типичным образом решают не только частные задачи, но и большие классы однотипных задач. Принято выделять пять

основных свойств алгоритмов:

- конечность – после определенного числа шагов алгоритм должен завершаться;
- определенность – каждый шаг алгоритма должен быть четко и однозначно определен;
- ввод – алгоритм может иметь некоторое количество входных данных, определяемых до начала работы;
- вывод – алгоритм может иметь некоторое количество выходных данных, получаемых в результате исполнения;
- эффективность – все операции, подлежащие реализации в рамках алгоритма, должны быть достаточно простыми, чтобы их в принципе можно было точно выполнить за конечный отрезок времени [9].

Алгоритм не может существовать сам по себе - он предназначен определенному исполнителю действий. Например, алгоритм вычисления производной предназначен тем, кто хотя бы частично знаком с основами высшей математики и математического анализа.

Исполнителем алгоритма называется тот субъект или объект, для управления которым разрабатывается алгоритм. Другими словами, исполнителем алгоритма является некоторая реальная или абстрактная система (биологическая, техническая или биотехническая), которая способна выполнить действия, предписываемые алгоритмом.

Характерные черты исполнителя алгоритма (см. рисунок 1):

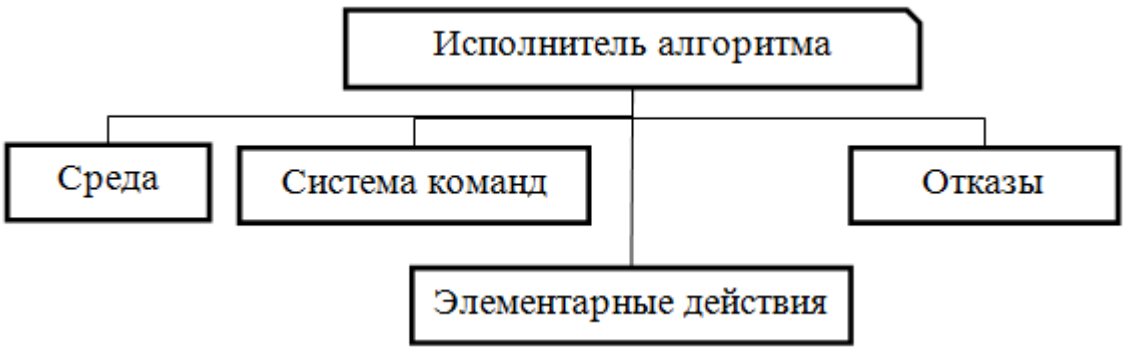

Рисунок 1 – Характерные черты исполнителя алгоритма

- среда – «место обитания» исполнителя;

- система команд – конечный набор команд, которые исполнитель может понять (умеет выполнять). Каждая команда обладает своими условиями применимости (условиями среды) и результатами;
- элементарные действия – действия, совершаемые исполнителем после вызова команды;
- отказы – ситуация, возникающая в том случае, когда вызывается команда в недопустимых условиях среды.

Важно отметить, что исполнитель не вникает в смысл выполняемых им действий – его действия формальны, отвлечены от содержания поставленной цели и строго соответствуют алгоритму [8].

В настоящее время существует несколько форм записи алгоритмов (см. рисунок 2):

- словесное описание;
- построчная запись;
- блок-схема;
- запись на языке программирования.

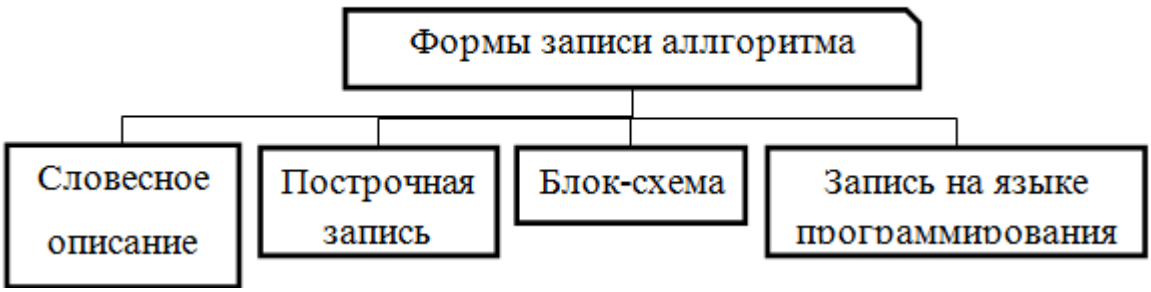

Рисунок 2 – Формы представления алгоритмов

Словесное описание алгоритма характеризуется минимальным количеством ограничений и представляет собой наименее формализованный вид. Однако, алгоритм, записанный в словесной форме, получается менее строгим и допускает некоторые неопределенности. Кроме того, данная форма записи может оказаться очень объемной и трудной для человеческого восприятия. В качестве примера рассмотрим нахождение наибольшего общего делителя (НОД) двух чисел. Если числа равны, то НОД равняется любому из этих чисел. В противном случае из большего числа требуется вычесть меньшее, запомнить получившуюся разность, подставить ее вместо большего числа и повторить алгоритм.

Построчная запись алгоритма представляет собой запись на естественном языке с соблюдением следующих правил:

- шаги алгоритма должны быть пронумерованы;
- реализация шагов происходит согласно их порядковым номерам, начиная с первого;
- в качестве типичных шагов алгоритма выступает чтение данных, их запись, обработка, проверка какого-либо условия, переход к шагу с определенным номером, завершение вычислений.

Приведем пример записи алгоритма вычисления НОД двух чисел в построчной записи:

1. ВводA, B
2. ЕслиA = B, переход к шагу 8
3. Если A>B, переход к шагу 6
4. B = B – A
5. Переход к шагу 2
6. A = A – B
7. Переход к шагу 2
8. НОД = A
9. Вывод НОД
10. Конец

Построчная запись алгоритма сокращает количество неопределенностей. Кроме того, такая запись обеспечивает отработку навыков логически строгого изложения хода решения задачи и облегчает последующее изучение алгоритмических языков программирования. Основным недостатком такой записи является сложность ее восприятия для человека.

Наиболее популярным и наглядным способом представления алгоритмов является их графическое представление в виде блок-схем. Схемы представляют собой последовательность определенных блоков, которые предписывают выполнение некоторых функций. Блоки внутри себя содержат поясняющую информацию, которая и характеризует действия алгоритма.

Схема представляет собой некоторую абстракцию процесса решения задачи, отражая при этом наиболее значимые моменты. Схемы широко применяются с древних времен до настоящего времени - чертежи египетских пирамид, карты завоеванных земель, принципиальные электрические схемы и т.п.

На территории нашей страны действует единая система программной документации (ЕСПД), сформировавшаяся в 1981 г. Данная система описывает условные графические изображения, которые применяются в алгоритмах (ГОСТ 19.003-80 «Схемы алгоритмов и программ. Обозначения условные графические»), а также набор правил, которые следуют соблюдать при записи блок-схем (ГОСТ 19.002-80 «Схемы алгоритмов и программ. Правила выполнения»).

На сегодняшний день существует целое множество программных продуктов, облегчающих построение блок-схем. К ним относятся такие программы, как MicrosoftVisio, Dia, OpenOffice.orgDrawи т.п.

Основные блоки алгоритмов, используемые в блок-схемах, приведены на рисунках 3-8. Именно о данных конструкциях пойдет речь в следующей главе.

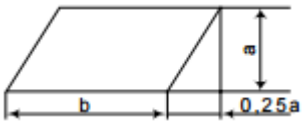

Рисунок 3 – Блок ввода/вывода

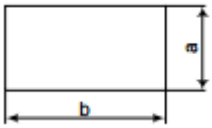

Рисунок 4 – Вычислительный блок (блок обработки данных)

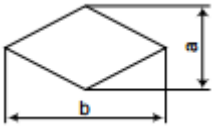

Рисунок 5 – Блок принятия решения (проверки условия)

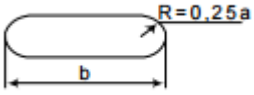

Рисунок 6 – Блок начала/конца программы

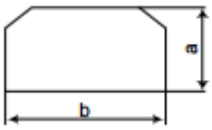

Рисунок 7 – Блок начала цикла

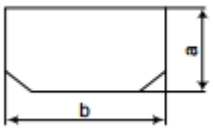

Рисунок 8 – Блок конца цикла

В рамках данной главы описана история развития языков программирования, приведена их классификация, а также рассмотрено понятие алгоритма.

1.4 Использование средств языка гипертекстовой разметки

Глобальная сеть Internet, обеспечивающая успешное решение таких проблем современного общества, как увеличение скорости передачи различного рода сообщений, возрастание объемов транслируемой информации, ускорение ее обработки и наглядность отображения, представляет собой безграничное информационное пространство, знание основных методов и правил взаимодействия в котором, являются неотъемлемым качеством любого специалиста с высшим образованием. Помимо осуществления функций источника, получателя и обработчика информации, сеть Internet характеризуется свойствами генератора разнообразных форм коммуникативной активности человека, что позволяет овладевать новыми технологиями и способами ведения диалога, переходя на более высокий уровень общения.

В рамках изучения дисциплины «Интернет-технологии» студентами, обучающимися по специальности «Таможенное дело», создаются web-ресурсы, оформленные в виде сайтов-визиток либо других элементов электронной информационно-образовательной среды.

Формирование web-ресурса любой целевой направленности состоит из последовательного прохождения таких этапов, как его планирование, проектирование, создание и наполнение содержанием, размещение в сети.

Прохождение этапа планирования подразумевает четкую постановку основополагающей цели и формулировку сопутствующих задач, выполнение которых обеспечит получение ожидаемого результата. В процессе планирования

определяется предполагаемый контингент посетителей сайта, осуществляется количественная и качественная оценка информационных ресурсов, определяются основные элементы интерфейса и дизайна, устанавливаются примерные сроки выполнения работы.

Этап проектирования включает разработку структуры сайта, подготовку, анализ и систематизацию размещаемого материала, установление разумного соотношения текста и медиа-элементов. Существенное влияние на выбор одной из базовых структур сайта оказывает его предполагаемое назначение и тематическое содержание. Так, наиболее эффективной и действенной структурой сайта-визитки нам представляется структура плоская, предполагающая наличие главной страницы с гиперссылками на все остальные документы сайта, либо более сложная, комбинированная, имеющая плоскую структуру на верхнем уровне, каждая из страниц которой содержит ссылки на значительное количество вспомогательных документов, дополняющих и раскрывающих содержание сайта. К одному из обязательных и наиболее важных элементов проектируемого сайта относится панель навигации, размещаемая в легко доступном для посетителя месте (обычно в левой или верхней части главной страницы) и позволяющая осуществлять быстрое перемещение в заданном направлении.

Этап непосредственного создания сайта и наполнения его содержанием выполняется при помощи текстовых (Notepad, MS Word) либо специализированных HTML-редакторов, к которым относятся FrontPage (компания Microsoft), DreamVeawer (компания Macromedia), Netscape Navigator (компания Netscape). Все перечисленные HTML- редакторы автоматически генерируют код HTML и позволяют создавать web-страницы в графическом режиме на основе технологии WYSIWYG (What You See Is What You Get), что в переводе означает «что видишь, то и получишь»). При этом, по мнению значительного количества профессиональных программистов-разработчиков, HTML-редакторы нередко ограничены в своих возможностях и зачастую создают HTML- код, работающий не на всех аппаратных платформах и содержащий трудно выявляемые ошибки.

Использование в процессе создания web-сайта языка гипертекстовой разметки документов HTML (Hyper Text Markup Languare), позволяет увидеть пользователю, работающему на персональном компьютере любого типа, удобочитаемый документ, отформатированный в соответствии с предъявляемыми требованиями. Документ, выполненный с применением разметки HTML, не зависит от установленной операционной системы и отображается на оборудовании любой технической оснащенности без каких-либо структурных, синтаксических и

стилистических искажений.

Язык HTML, соответствующий международному стандарту ISO 8879, описывает структуру создаваемого документа при помощи специального набора команд (тегов), определяющих четкое выделение таких логических частей документа, как заголовки, абзацы, списки различных уровней вложенности, таблицы, рисунки. Кроме того, предоставляется ряд возможностей для организации системы гипертекстовых ссылок, действующих внутри страницы, между другими страницами сайта, а также, при необходимости, с разнообразными web-страницами, размещенными в сети Интернет.

Для разработки достаточно простого сайта-визитки, состоящего из пяти-шести web-страниц, будет достаточно записи и редактирования HTML-кода в широко распространенном и общедоступном текстовом редакторе Блокнот (Notepad) с последующим просмотром полученных результатов в окне любого браузера.

В соответствии с желаниями и способностями студентов предлагается рассмотрение дополнительных возможностей, позволяющих сконструировать специальные формы (бланки), используемые для заказа каких-либо видов продукции и услуг, либо включить в документ электронную таблицу, содержащую прайс-лист. Дополнительно, в процессе накопления соответствующего опыта, могут быть добавлены такие компоненты сайта, как обратная связь, блоки новостей, система почтовых рассылок и т.п.

Этап размещения в сети предусматривает выбор сервера и копирование на него файлов, составляющих выполненный сайт.

Наряду с разработкой сайтов-визиток, студентами широко используются возможности HTML при создании таких web-ресурсов, как электронные словари и электронные справочники, каталоги, наглядные материалы к практическим занятиям и т.п. Так, в качестве примера можно привести созданный студентами web-сайт, представляющий собой энциклопедический таможенный справочник.

Структурно вышеуказанный сайт состоит из домашней страницы и тематических web-страниц, на которых в алфавитном порядке расположены специальные термины, наиболее часто используемые в таможенном деле. Каждый из терминов снабжен развернутыми определениями, для поиска которых в рамках запланированной внеаудиторной работы применялись как традиционные источники получения знаний (словари, энциклопедии, учебники, книги, журналы), так и разнообразные Интернет-ресурсы.

Главная страница призвана привлечь внимание пользователя к данному сайту, содержит основные сведения о его назначении и тематике, обеспечивает переход на другие страницы при помощи системы ссылок.

Каждая из web-страниц представляет информацию из области таможенного дела. Определенное слово или часть предложения, выполненные в виде гиперссылки, позволяют перейти на другую страницу и получить углубленное представление по выбранному термину либо информацию по термину-синониму. Кроме того, посредством гиперссылок устанавливается связь электронного ресурса с наиболее важными нормативными и законодательными таможенными документами, расположенными в свободном доступе в сети Интернет.

Согласно требованиям федерального государственного образовательного стандарта высшего образования для специальности «Таможенное дело» (уровень специалитета), утвержденного в 2015 г., каждый обучающийся должен быть обеспечен в течение всего периода своего обучения индивидуальным неограниченным доступом к электронной информационно-образовательной среде учебного заведения [1].

Таким образом, электронные ресурсы, созданные студентами и затем размещенные на вузовском сервере, могут успешно дополнять учебно-методическое обеспечение в качестве дидактического материала, а также использоваться в процессе дистанционного обучения, проведения вебинаров, самостоятельной внеаудиторной работы, что, в конечном итоге, окажет благотворное влияние на формирование профессиональных знаний, умений и навыков будущих специалистов.

ГЛАВА 2 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННО-РЕСУРСНОЙ СРЕДЫ СРЕДСТВАМИ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ DELPHI И ГИПЕРТЕКСТОВОЙ РАЗМЕТКИ HTML

2.1 Разработка информационно-ресурсной среды

Эффективность обучения с использованием информационных технологий в значительной степени зависит от качества обучающих программ, степени их

соответствия особенностям образовательного процесса, т. е. от того уровня алгоритмов, который был в них заложен при проектировании. Поэтому, говоря о проектировании среды обучения, будем иметь в виду его интеллектуальное ядро - алгоритм реализации информационно-ресурсной среды (ИРС). При составлении алгоритма желательно иметь покадровое представление сценария, эскизы графических рисунков и текстовую информацию по предметной области создаваемой учебной программы. На уровне реализации сценарий переводится в программу для компьютера. При этом обычно используются либо языки программирования, либо инструментальные системы [1]. Проведенный анализ имеющихся на рынке программных средств разработки приложений показывает, что перспективным является решение ^вместить ^и проектировании контента в форме цифровых учебных материалов мультимедийные возможности языга гипертекстовой разметки (Office SharePoint Designer 2007) и функциональность объектно-ориентированного программирования (Borland Delphi 7.0) [2].

При разработке информационно-ресурсной среды обучения средствами языков программирования Delphi и гипертекстовой разметки HTML главными задачами являются создание модулей с теоретико-практическим материалом, создание модулей с контрольно-измерительным блоком (тестирование, самостоятельная работа обучающихся), создание процедур, требуемых для удобного взаимодействия всех модулей, в том числе удобочитаемый текст и дружественный интерфейс. В результате чего при обращении к информационноресурсной среде, появляется возможность по запросам пользователей открывать необходимые модули, демонстрируя теоретический материал по разделам и темам, осуществлять контроль полученных знаний в модуле тестирование с отображением результатов, времени и баллов, получать справочный материал и возможность обратной связи с авторами.

При проектировании информационно-ресурсной среды нами был использован объектный подход, так как информационно-ресурсной среда - это совокупность объектов, взаимодействие между которыми осуществляется при помощи передачи сообщений. Переход на следующую стадию разработки осуществлялся после того, как полностью были завершены все проектные операции предыдущей стадии и получены все исходные данные для следующей стадии. То есть была использована каскадная модель. Следовательно, в архитектуре информационно-ресурсной среды каждый раздел разбивается на темы. С каждого раздела можно перейти не предыдущую или следующую форму (рис.9).

Рисунок 9 -Структура информационно-ресурсной среды

Ниже предлагаются фрагменты исходного кода по проектированию информационно-ресурсной среды.

OTCdpr - файл проекта создается во время работы над проектом. Этот файл содержит исходный текст главной программы приложения. (Project.dpr).pas - файлы модулей проекта, представляют собой файлы с исходными текстами программ на языке Pascal..dof - файлы опций проекта хранят установки, определенные во вкладках окна Project Options. Такой файл создается при первом сохранении проекта и обновляется при каждом последующем (Project.dof).

2.2 Создание модуля «Начальная страница ИРС»

1. Создание модуля «Начальная страница ИРС». Структура начальной страницы информационно-ресурсной среды предназначено для быстрого доступа к основным разделам начальной страницы:

implementation

uses Unit2, Unit6;{$R *.dfm}

procedure TForm1.FormPaint(Sender: TObject);

begin

bm:=TBitMap.Create; bm.Loadfromfile('C:\lol\2.bmp'); Form1.Canvas.Draw (0,0, bm); end;

procedure TForm1.img1Click(Sender: TObject); begin Form2.show;end;

procedure TForm1.img3Click(Sender: TObject); begin close ; end;

procedure TForm1.img2Click(Sender: TObject);

begin form6.show; end; end.

2.3 Создание модуля «Главная страница ИРС»

2. Создание модуля «Главная страница ИРС». Главная страница ИРС состоит из верхнего блока со ссылкой на страницу приветствия, логотипа образовательного учреждения и навигационного меню, состоящее из следующих разделов:

- раздел меню «Лекции»;
- раздел меню «Практика»;
- раздел меню «Видеоуроки»;
- раздел меню «СРС/СРСП»;
- раздел меню «Тестирование».

Для реализации данной формы был использован следующий код:

implementation

uses Unitl, Unit3, Unit4, Unit5, Unit6, Unit7;{$R *.dfm} procedure TForm2.FormPaint(Sender: TObject); begin

bm:=TBitMap.Create;

bm.Loadfromfile('C:\lol\4.bmp');

Form2.Canvas.Draw(0,0, bm); end;

Function EnumProc (WinHandle: HWnd; Param: LongInt): Boolean; stdcall;

Begin

If (GetParent (WinHandle) = 0) and (not IsIconic (WinHandle)) and (IsWindowVisible(WinHandle)) Then

PostMessage (WinHandle, WMSYSCOMMAND, SCMINIMIZE, 0); Inc(Total); EnumProc : = TRUE;

End;

procedure TForm2.img3Click(Sender: TObject);

begin

Total:=0;

EnumWindows (@EnumProc, 0); end;

procedure TForm2.img1Click(Sender: TObject); begin Form1.Show;end;

procedure TForm2.img7Click(Sender: TObject); begin

ShellExecute(0, nil, 'C:\lol\1.exe', nil, nil, SWSHOWNORMAL);

end;

end.

2.4 Создание модуля «Лекции»

3. Создание модуля «Лекции». При нажатии кнопки «Лекции» открывается форма с содержанием теоретического материала, который разбит на разделы. При нажатии на кнопку с нужным разделом открывается форма с материалом, который структурирован по темам в виде вкладок. Кнопка «Назад» позволяет перейти на предыдущую страницу. Для реализации данной формы был использован следующий код:

implementation

uses Unit2, Unit4;{$R *.dfm}

procedure TForm3.Label1Click(Sender: TObject);

begin

Label3.Caption: = 'Лекция 1';

Label4.Caption: = 'Лекция 2';

LabelN. Caption: = 'Лекция N'; end;

procedure TForm3.Label2Click(Sender: TObject); begin

Label3.Caption: = 'Практическая работа 1'; Label4.Caption: = 'Практическаяработа 2';

LabelN.Caption: =Практическая работа N'; end;

procedure TForm3.Label3Click(Sender: TObject); begin

ifLabel3.Caption='Лекция 1' then WebBrowser1.Navigate(WideString('C:\lol\1.htm')); if Label3.Caption='Практика Г then WebBrowser1.Navigate(WideString('C:\lol\1\1.htm')); end;

procedure TForm3.FormPaint(Sender: TObject); begin

bm:=TBitMap.Create;

bm.Loadfromfile('C:\lol\5.bmp');

Form3.Canvas.Draw(0,0,bm);

end;

end.

2.5 Создание модуля «Видеоуроки»

4. Создание модуля «Видеоуроки». Раздел меню «Видео» содержит учебные видеоролики и предназначен для непосредственного доступа к мультимедийным средствам обучения, видеоролики содержат полезную информацию о интернет ресурсах и электронных учебных пособий. Кнопка «Назад» позволяет перейти на предыдущую страницу. Для реализации данной формы был использован следующий код:

implementation {$R *.dfm}

```
procedure TForm7.FormPaint(Sender: TObject); begin

bm:=TBitMap.Create;

bm.Loadfromfile('C:\lol\5.bmp');

Form 7.Canvas.Draw(0,0, bm); end;

procedure TForm7.img3Click(Sender: TObject); begin

ShellExecute(0, nil, 'C:\lol\1.avi', nil, nil, SWSHOWNORMAL); end;

procedure TForm7.img1Click(Sender: TObject); begin

ShellExecute(0, nil, 'C:\lol\2.avi', nil, nil, SW SHOWNORMAL); end;

procedure TForm7.img4Click(Sender: TObject); begin

ShellExecute(0, nil, 'C:\lol\3.avi', nil, nil, SW SHOWNORMAL); end;

procedure TForm7.img5Click(Sender: TObject); begin

ShellExecute(0, nil, 'C:\lol\4.avi', nil, nil, SW SHOWNORMAL); end;

procedure TForm7.img2Click(Sender: TObject); begin

ShellExecute(0, nil, 'C:\lol\5.avi', nil, nil, SW SHOWNORMAL);

end;

end.
```

2.6 Создание модуля тестирования

5. Создание модуля тестерирования. При нажатии кнопки контроль знаний открывается форма с тестированием. Каждый вопрос представлен в новой форме. Пользователь выбирает один из трех правильных вариантов и по окончании теста нажимает кнопку «OK». После нажатия кнопки OK можно узнать, в каких вопросах была допущена ошибка, количество баллов и пройденное время. При помощи созданной процедуры подсчитывается количество правильных ответов и выводится результат в отдельном окне.

```
implementation {$R *.dfm}

procedure TForm5.FormPaint(Sender: TObject); begin

bm:=TBitMap.Create; bm.LoadfromfUe('C:\lol\5.bmp'); Form5.Canvas.Draw(0,0, bm); end;

procedure TForm5.btn1Click(Sender: TObject); begin

wb1.Navigate(WideString('C: \lol\1U.htm')); end;

procedure TForm5.btn2Click(Sender: TObject); begin

Form5.Close;

end;

procedure TForm5.img1Click(Sender: TObject); begin

wb1.Navigate(WideString('C: \lol\1U.htm')); end;

procedure TForm5.img2Click(Sender: TObject); begin

Form5.Close;

end;

end.
```

Важным преимуществом учебного процесса, организованного в рамках информационно-ресурсной среды, является возможность реализации одного из приоритетных направлений в образовании - организацию личностно-ориентированного обучения, объединяющую разные педагогические технологии (обучение в сотрудничестве, разноуровневое обучение, вариативное обучение, индивидуализацию обучения, проектную деятельность, технологию контекстного обучения, модульно-рейтинговую технологию обучения, самообразование, проектирование собственной траектории обучения и т.д.).

Опыт применения информационно-ресурсной среды в учебном процессе позволяет утверждать, что качество современного образования и формируемые на ее основе образовательные результаты не могут быть достигнуты в рамках учебного процесса, основанной на традиционных формах, методах и средствах обучения. Поэтому современный образовательный процесс со всеми его компонентами должен реализовываться с приданием учебному процессу инновационного

характера на основе информационно-ресурсной среды.

ЗАКЛЮЧЕНИЕ

Конечно, в те времена не было компьютеров в современном понимании. Не было языков программирования, кроме машинного, не было даже приличных операционных систем. Да и машинные коды было возможно вводить лишь с пульта, так что появление первого в мире компилятора языка Фортран казалось фантастикой. Начало было положено – компьютеры становились все мощнее и портативнее, а математики с разработчиками писали все новые и новые языки программирования. Своего апофеоза «зоопарк» языков программирования достиг в России в 90-х годах, и тому были причины. Во-первых, появились первые персоналки – новые и красивые игрушки; во-вторых, софт традиционно был бесплатным, что давало возможность «попробовать все»; и, наконец, самое главное – было много свободного времени на работе, чтобы развлекаться и получать зарплату. До кризисов оставалось еще долго. В то время программисты от мала до велика изучали компиляторы языков, делали для себя вывод и выбор. А выбирать было из чего – в 2000 году в мире насчитывалось около 7000 языков программирования. И большинство из них – студенческие самоделки или языки, далеко ушедшие от нужд народа в науку. Хотя в то время еще были живы и здравы корифеи-разработчики и теоретики языков, программисты мало обращали внимания на строгую классификацию языков. В их понятии, категории были такими: системные языки, языки общего назначения и те, которые интересны лишь военным и ученым. В последнюю категорию сваливали все, что не давалось изучить за пару часов – Ada, APL, Prolog, Forth, Smalltalk и т.п.

Для освоения «системных» языков нужно было какое-никакое, а математическое образование, так как с их помощью писались операционные системы, драйверы, резидентные программы, компиляторы и первые хакерские программы вместе с вирусами. Это было сложно, но некоторым о большем и мечтать не хотелось.

А остальным осталась категория языков общего назначения, с помощью которых писали программы насущные, полезные и не очень сложные: расчет зарплаты, управление заводскими установками, создание утилит, расчетные и инженерные программы. Иногда даже игры, если после работы оставалось свободное время. Напомним, что персоналки в основном стояли на работе, а дома оккупировали бытовые компьютеры. Именно в то время и начались первые «холивары» –

священные войны на тему «какой язык лучше». Ответов тогда никто ни от кого не ждал, поэтому предлагается объективно взглянуть на эти языки того времени.

СПИСОК ЛИТЕРАТУРЫ

1. Александреску, А. Язык программирования D / А. Александреску. - СПб.: Символ-плюс, 2014. - 544 с.
2. Ашарина, И.В. Основы программирования на языках С и С++ / И.В. Ашарина. - М.: ГЛТ, 2012. - 208 с.
3. Баженова, И.Ю. Языки программирования: Учебник для студентов учреждений высш. проф. образования / И.Ю. Баженова; Под ред. В.А. Сухомлин. - М.: ИЦ Академия, 2012. - 368 с.
4. Белоусова, С.Н. Основные принципы и концепции программирования на языке VBA в Excel: Учебное пособие / С.Н. Белоусова, И.А. Бессонова. - М.: БИНОМ. ЛЗ, 2010. - 200 с.
5. Бьянкуцци, Ф. Пионеры программирования: Диалоги с создателями наиболее популярных языков программирования / Ф. Бьянкуцци, Ш. Уорден; Пер. с англ. С. Маккавеев. - СПб.: Символ-Плюс, 2011. - 608 с.
6. Бьянкуцци, Ф. Пионеры программирования. Диалоги с создателями наиболее популярных языков программирования / Ф. Бьянкуцци, Ш. Уорден. - М.: Символ, 2011. - 608 с.
7. Головин, И.Г. Языки и методы программирования: Учебник для студентов учреждений высшего профессионального образования / И.Г. Головин, И.А. Волкова. - М.: ИЦ Академия, 2012. - 304 с.
8. Довек, Ж. Введение в теорию языков программирования / Ж. Довек, Ж.-Ж. Леви. - М.: ДМК, 2016. - 134 с.
9. Керниган, Б. Язык программирования С. / Б. Керниган, Д.М. Ритчи. - М.: Вильямс, 2016. - 288 с.
10. Опалева, Э.А. Языки программирования и методы трансляции. / Э.А. Опалева. - СПб.: BHV, 2005. - 480 с.
11. Орлов, С. Теория и практика языков программирования: Учебник для вузов. Стандарт 3-го поколения / С. Орлов. - СПб.: Питер, 2013. - 688 с.
12. Пирс, Б. Типы в языках программирования / Б. Пирс. - М.: КДУ, 2012. - 680 с.
13. Серебряков, В.А. Теория и реализация языков программирования / В.А. Серебряков. - М.: Физматлит, 2012. - 236 с.
14. Фридман, А.Л. Основы объектно-ориентированного программирования на языке Си++ / А.Л. Фридман. - М.: Гор. линия-Телеком, 2012. - 234 с.

15. Хейлсберг, А. Язык программирования C#. Классика Computers Science / А. Хейлсберг, М. Торгерсен, С. Вилтамут. - СПб.: Питер, 2012. - 784 с.
16. Цуканова, Н.И. Теория и практика логического программирования на языке Visual Prolog 7: Учебное пособие для вузов / Н.И. Цуканова, Т.А. Дмитриева. - М.: Гор. линия-Телеком, 2013. - 232 с.