Федеральное государственное бюджетное образовательное учреждение
высшего образования «Кемеровский государственный университет»
Новокузнецкий институт (филиал)

Факультет информатики, математики и экономики
Кафедра информатики и вычислительной техники им. В. К. Буторина

О. А. Штейнбрехер

Введение в профессиональную деятельность

Методические указания по организации самостоятельной работы по дисциплине «Введение в профессиональную деятельность» для обучающихся по направлению подготовки 09.03.03 Прикладная информатика Профиль «Прикладная информатика в экономике»

Новокузнецк
2020

УДК [378.147.88:004.4](072)
ББК 74.484(2Рос-4Кем)я73+ 32.972я73
Ш88

Ш88 «Введение в профессиональную деятельность. Методические указания по организации самостоятельной работы» : метод. указ (текст. электрон. изд.)/ О.А. Штейнбрехер ; Новокузнец. ин-т (фил.) Кемеров. гос. ун-та – Новокузнецк: НФИ КемГУ, 2020. – 18 с.

Приводятся методические указания по организации самостоятельной работы по дисциплине «Введение в профессиональную деятельности».

Методические указания предназначены для студентов всех форм обучения направлений 09.03.03 «Прикладная информатика».

Рекомендовано
на заседании кафедры
информатики и вычислительной
техники им. В. К. Буторина
23 ноября 2020 года.
Заведующий кафедрой

А. В. Маркидонов

Утверждено
методической комиссией факультета
информатики, математики и экономики
17 декабря 2020 года.
Председатель методкомиссии

Г.Н. Бойченко

УДК [378.147.88:004.4](072)
ББК 74.484(2Рос-4Кем)я73+ 32.972я7

Текст представлен в авторской редакции

Оглавление

Классификация автоматизированных информационных систем по отраслям.

Краткие теоретические сведения

Система программ «1С: Предприятие 8» включает в себя платформу и прикладные решения, разработанные на ее основе, для автоматизации деятельности организаций и частных лиц. Сама платформа не является программным продуктом для использования конечными пользователями, которые обычно работают с одним из многих прикладных решений (конфигураций), разработанных на данной платформе. Такой подход позволяет автоматизировать различные виды деятельности, используя единую технологическую платформу.

Платформа 1С: Предприятие обладает следующими стандартными видами объектов: константы, перечисления, справочники, документы, регистры и т.д. Все объекты конфигурации имеют свойство «*Имя*» и «*Синоним*». «*Имя*» объекта имеет ограничения: оно не должно содержать пробелы и не должно начинаться с символа или цифры. «*Синоним*» отображается в режиме пользователя на формах.

Сложные объекты, такие как справочники, документы, регистры и так далее имеют поля – реквизиты. Рассматривая структуру конфигурации как базу данных – объекты базы являются таблицами базы, а реквизиты объектов – полями таблицы. При этом константы и перечисления являются таблицами, состоящими из одного поля. Реквизиты объектов определяются пользователем, но каждый объект имеет несколько стандартных реквизитов.

Формы в 1С:Предприятие предназначены для отображения и редактирования информации, содержащейся в базе данных. Формы могут принадлежать конкретным объектам конфигурации или существовать отдельно от них и использоваться всем прикладным решением в целом. Каждый объект конфигурации может использоваться для выполнения некоторых стандартных действий. Например, для любого справочника может

потребоваться отображать список его элементов, отображать отдельные элементы справочника, отображать группу справочника, выбирать элементы и группы элементов из справочника. Для любого документа список таких действий будет гораздо меньше: просмотр списка документов, выбор из списка документов и просмотр отдельного документа.

Важной особенностью системы 1С:Предприятие 8 является механизм автогенерируемых форм. Этот механизм освобождает разработчика от необходимости создания всех возможных форм для каждого из объектов конфигурации. Разработчику достаточно добавить новый объект конфигурации, а система сама сгенерирует в нужные моменты работы пользователя необходимые формы для отображения информации, содержащейся в этом объекте.

Таким образом, разработчику нужно создавать собственные формы объектов прикладного решения лишь в том случае, если они должны иметь отличия (другой дизайн или специфическое поведение) от форм, автоматически генерируемых системой.

Принадлежность формы тому или иному объекту конфигурации не определяет состав данных, которые отображаются в форме. То, что форма принадлежит, например, справочнику Номенклатура, позволяет назначить ее одной из основных форм для этого справочника, но никак не определяет, какие же именно данные будет отображать эта форма, и каково будет ее поведение.

Для того чтобы связать форму с данными, используются реквизиты формы, в которых указывается перечень данных, отображаемых формой. Все формы, сами по себе, имеют одинаковое поведение, независимо от того, какие данные они отображают. Однако один из реквизитов формы может быть назначен для нее основным (он выделяется жирным шрифтом), и в этом случае стандартное поведение формы и ее свойства будут дополнены в зависимости от того, какой тип имеет основной реквизит формы.

Управляемое приложение поддерживает следующие типы клиентов:

- толстый клиент (обычный и управляемый режим запуска)
- тонкий клиент
- веб-клиент

В управляемом приложении используются формы, построенные на новой технологии. Они называются Управляемые формы. Для облегчения перехода прежние формы (т.н. Обычные формы) также поддерживаются, но их функциональность не развивается и они доступны только в режиме запуска толстого клиента.

Основная особенность форм заключается в том, что они не нарисованы разработчиком детально, «по пикселям». Форма в конфигурации представляет собой логическое описание состава формы. А конкретное размещение элементов выполняется системой автоматически при отображении формы.

Редактор формы используется для создания и редактирования форм объектов прикладного решения. Формы объектов используются системой для визуального отображения данных в процессе работы пользователя.

Любая форма представляет совокупность нескольких составляющих:

- элементов — объектов, определяющих визуальное представление формы и осуществляющих взаимодействие с пользователем,
- командного интерфейса — совокупности команд, отображаемых в форме;
- реквизитов — объектов, данные которых форма использует в своей работе.
- команд — действий, которые определены в данной конкретной форме,
- параметров — объектов, значения которых характеризуют саму форму, используются при ее создании и остаются постоянными в процессе «жизни» формы,
- модуля — программы на встроенном языке, отвечающей за работу с элементами и за обработку событий.

Индивидуальная работа №1

1. Поставить в соответствие объекты конфигурации (справочники, документы, перечисления) и объекты демонстрационной базы. Заполнить таблицу.

Объект конфигурации	Пример	Назначение, реквизиты
Справочники		
Документы		
Отчет		
Перечисление		

2. Настроить фильтрацию и отсортировать форму элемента списка справочника.

3. Настроить панели разделов (добавить или удалить разделы в отображении).

4. Добавить на рабочий стол отчеты в левую колонку. Добавить полнотекстовый поиск в правую колонку.

5. Изменить настройки формы элемента справочника Контрагенты. Изменить группировку Контактов на вертикальную. Выделить поля реквизитов и т.д. Увеличить размер шрифта.

6. Настроить сортировку данных в отчете. Настроить отображение отчета.

7. Отменить проведение нескольких документов. Найти отражение изменений в соответствующем отчете.

8. Составление штатного расписания

1. Установить дату 01.02

2. На установленную дату составлено штатное расписание.

а) В справочник Подразделения добавить следующие подразделения:

администрация

отдел Кадров

технический отдел

b) Справочник должности дополнить следующими должностями:

директор

бухгалтер

секретарь

менеджер отдела кадров

системный администратор

программист

c) Ввести штатное расписание

9. Оформление принятых на работы сотрудников. С 01.02принять на работу и ввести персональные данные сотрудников. Например:

Администрация Семенов Павел Петрович 15.10.1969 По месячной тарифной ставке 45000 р.

Оклад по дням Предъявлен паспорт. Женат, гр. России, ИНН 780615892620 и номер в ПРФ 070-469-985 93, сын 2004 г.р., образование высшее, диплом предъявлен.

Стаж 20 лет, 10 мес 10 дней,

адрес ул. 193318, ул. Белышева д.18, кв.23

11. Рассчитать зарплату за февраль (вкладка Расчет зарплаты - Начисление зарплаты).

12. Рассчитать налоги за февраль (вкладка Налоги – Журнал расчета страховых взносов в ПФР, ФОМС, и ФСС).

13. Произвести выплату зарплаты за февраль (вкладка Расчет зарплаты – Зарплата к выплате, способ выплаты – через кассу, выплачивать зарплату).

14. На основании платежной ведомости оформить расходный кассовый ордер. (в журнале «Зарплата к выплате» выделить документ зарплата за февраль – кнопка Действие – На основании – ПКО, ввести № 1).

15. Сформировать расчетные листки и расчетную ведомость за февраль (вкладка Расчет зарплаты – группа Отчеты – Расчетные листки(Расчетная ведомость)).

16. Установить дату последнее число марта месяца.

Выполненные задания оформить в виде отчета с описанием проделанных действий и иллюстрациями в виде копий экрана.

Классификация языков программирования и средств разработки.

Краткие теоретические сведения

Функциона́льное программи́рование — раздел дискретной математики и парадигма программирования, в которой процесс вычисления трактуется как вычисление значений функций в математическом понимании последних (в отличие от функций как подпрограмм в процедурном программировании).

Противопоставляется парадигме императивного программирования, которая описывает процесс вычислений как последовательное изменение состояний (в значении, подобном таковому в теории автоматов). При необходимости, в функциональном программировании вся совокупность последовательных состояний вычислительного процесса представляется явным образом, например, как список.

Функциональное программирование предполагает обходиться вычислением результатов функций от исходных данных и результатов других функций, и не предполагает явного хранения состояния программы. Соответственно, не предполагает оно и изменяемость этого состояния (в отличие от императивного, где одной из базовых концепций является переменная, хранящая своё значение и позволяющая менять его по мере выполнения алгоритма).

Наиболее известными языками функционального программирования являются:

- Лисп — (Джон Маккарти, 1958) и множество его диалектов;
- Erlang — (Joe Armstrong, 1986) функциональный язык с поддержкой процессов;
- APL — предшественник современных научных вычислительных сред, таких как MATLAB;
- ML (Робин Милнер, 1979, из ныне используемых диалектов известны Standard ML и Objective CAML);

- F# — функциональный язык семейства ML для платформы .NET;
- Scala;
- Miranda (Дэвид Тёрнер, 1985, который впоследствии дал развитие языку Haskell);
- Nemerle — гибридный функционально/императивный язык;
- XSLT и XQuery;
- Haskell — чистый функциональный.

Рассмотрим язык программирования F#.

Объявление и инициализация происходит одновременно. Изменить значение нельзя.

```
let x = 1
```

```
let x = 1
x = x + 1 // Это выражение ничего не присваивает!
```

Задание функций. Работа со списками.

```
// Определим функцию, которая вычисляет квадрат значения
let square x = x * x
let getSquares items = items |> List.map square
let lst = [ 1; 2; 3; 4; 5 ] // Создать список в F#
printfn "Квадрат числа %A равен %A" lst (getSquares lst)

// Создадим список квадратов первых 100 натуральных чисел
let first100Squares = [ for x in 1..100 -> x * x ]
// То же самое, но массив!
let first100SquaresArray = [| for x in 1..100 -> x * x |]
// Функция, которая генерирует бесконечную последовательность нечетных чисел
//
// Вызывать вместе с Seq.take!
let odds =
```

```
    let rec loop x = // Использует рекурсивную локальную функцию
    seq { yield x yield! loop (x + 2) }
    loop 1
printfn "Первые 3 нечетных числа: %A" (Seq.take 3 odds)
// Вывод: "Первые 3 нечетных числа: seq [1; 3; 5]
```

Индивидуальная работа №2

1. Реализуйте вычисление корней квадратного уравнения на F#
2. Реализуйте функцию вычисления кубов последовательности
3. Реализуйте вывод геометрической последовательности
4. Реализуйте вычисление факториала, с использование цикла и рекурсивной функции

Классификация прикладного программного обеспечения по сферам деятельности.

Краткие теоретические сведения

LaTeX (произносится /ˈlɑːtɛх/ или /ˈleɪtɛх/) — наиболее популярный набор макрорасширений (или макропакет) системы компьютерной вёрстки TeX, который облегчает набор сложных документов. В типографском наборе системы TeX форматируется традиционно как LATEX.

Пакет позволяет автоматизировать многие задачи набора текста и подготовки статей, включая набор текста на нескольких языках, нумерацию разделов и формул, перекрёстные ссылки, размещение иллюстраций и таблиц на странице, ведение библиографии и др.

Общий внешний вид документа в LaTeX определяется стилевым файлом. Существует несколько стандартных стилевых файлов для статей, книг, писем и т. д., кроме того, многие издательства и журналы предоставляют свои собственные стилевые файлы, что позволяет быстро оформить публикацию, соответствующую стандартам издания.

Во многих развитых компьютерных аналитических системах, например, Maple, Mathematica, Maxima, Reduce возможен экспорт документов в формат *.tex.

Возможности системы, в принципе, не ограничены (благодаря механизму программирования новых макросов). Вот список некоторых возможностей:

- алгоритмы расстановки переносов, определения междусловных пробелов, балансировки текста в абзацах;
- автоматическая генерация содержания, списка иллюстраций, таблиц и т. д.;
- механизм работы с перекрёстными ссылками на формулы, таблицы, иллюстрации, их номер или страницу;

- механизм цитирования библиографических источников, работы с библиографическими картотеками;
- размещение иллюстраций (иллюстрации, таблицы и подписи к ним автоматически размещаются на странице и нумеруются);
- оформление математических формул, возможность набирать многострочные формулы, большой выбор математических символов;
- оформление химических формул и структурных схем молекул органической и неорганической химии;
- оформление графов, схем, диаграмм, синтаксических графов;
- оформление алгоритмов, исходных текстов программ (которые могут включаться в текст непосредственно из своих файлов) с синтаксической подсветкой;
- разбивка документа на отдельные части (тематические карты).

Документ LaTeX — это текстовый файл, содержащий специальные команды языка разметки (рисунок 1).

Процесс создания документов с системе LaTeX состоит из следующих этапов:

1. В LaTeX-редакторе создать исходный файл (LaTeX-файл) – файл с расширением .tex (например, hi.tex), который содержит текст документа и специальные команды, указывающие LaTeX, как именно нужно сверстать этот текст.
2. Скомпилировать исходный файл (hi.tex) в файл документа, например в формате PDF (hi.pdf) с помощью PDFLaTeX или XeLaTeX.
3. Посмотреть результирующий файл. Если результат устраивает, распечатать его. Иначе внести изменения в исходный файл, снова скомпилировать его и т.д.

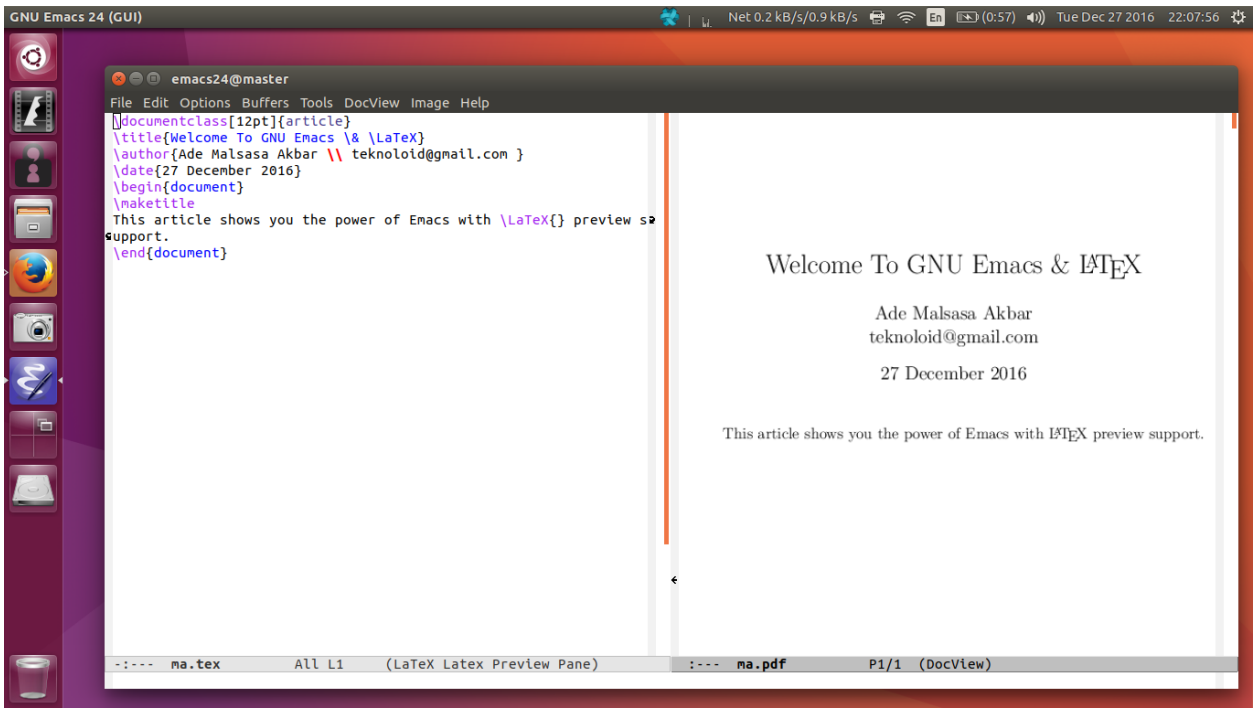

Рисунок 1 – Пример документа LaTeX

Сам документ делится на преамбулу и тело. Преамбула содержит информацию про класс документа, использованные пакеты макросов, определения макросов, автора, дату создания документа и другую информацию.

Тело документа содержит собственно текст документа и команды разметки. Оно должно находиться между командами \begin{document} и \end{document}.

Символы пробела и табуляции, а также одиночный разрыв строки, рассматриваются LaTeX как символ пробела. Несколько последовательных символов пробела или табуляции рассматриваются как один символ. Пустая строка между двумя строками текста разделяет абзацы. Несколько пустых строк также интерпретируются как одна.

Следующие символы зарезервированы и имеют особое значение для LaTeX или недоступны во всех шрифтах. Для использования этих символов следует добавить обратную косую черту перед символом (или записать его в виде команды, начинающейся с \):

- \# или **\textnumbersign** для #;

- **\$** или **\textdollar** для $;
- **\%** или **\textpercent** для %;
- **\textasciicircum** для ^;
- **\&** или **\textampersand** для &;
- **\{** и **\}** для { и }, соответственно;
- **\textasciitilde** для ~;
- **\backslash** или **\textbackslash** для \.

Ниже представлены команды для представления математических символов:

^ - верхний индекс (индексы в два и более символа надо заключать в фигурные скобки);

_ - нижний индекс;

\frac{dX_i}{dt} - деление: в первых скобках числитель, во вторых скобках знаменатель;

\ldots … - многоточие;

\longrightarrow - длинная стрелка слева направо;

\bar - верхняя черта над символом;

\dot - точка над символом;

\sum - большой знак суммы;

\prod - большой знак произведения;

\ - пробел;

\cdot - знак умножения в виде точки;

\times - знак умножения крест;

\int_{lower}^{upper} - интеграл.

Команды LaTeX чувствительны к регистру и существуют в следующих вариантах:

- обратная косая черта \ и алфавитная последовательность (например: \textampersand); имя команды завершается первым неалфавитным символом (пробелом, цифрой, или иным); отметим, что пробелы после имени команды, как правило, игнорируются;

- обратная косая черта и один неалфавитный символ; многие сочетания с символами используются для создания акцентов или спецсимволов; многие такие команды используются для ввода особых символов LaTeX и расстановки диакритических знаков;
- активный символ — такой, как тильда ~, используемая для обозначения неразрывного пробела.

Некоторым командам необходим аргумент, который, как правило, передается в фигурных скобках { } после имени команды. (Фигурные скобки допустимо опустить, если аргументом является один ASCII-символ, как в случае \nicefrac12 для ½.) Некоторые команды поддерживают необязательные параметры, указываемые после имени команды в квадратных скобках [ ]. Кроме того, существуют команды, аргументы которым передаются в круглых скобках, или иным образом.

Общий синтаксис следующий:

\имя_команды[параметр1][параметр2,...]{аргумент1}{аргумент2}...

На рисунке 2 представлены команды для отображения скобок.

Symbol	LaTeX	Comment	Symbol	LaTeX	Comment
$\vert$	`\vert`	divides	$\Vert$	`\\|`	divides unitarily, is parallel with
$($	`( \,`	left parenthesis	$)$	`) \,`	right parenthesis
$\{$	`\{`	left brace	$\}$	`\}`	right brace
$\lceil$	`\lceil`	ceiling (left)	$\rceil$	`\rceil`	ceiling (right)
$\ulcorner$	`\ulcorner`		$\urcorner$	`\urcorner`	

Symbol	LaTeX	Comment	Symbol	LaTeX	Comment
$/$	`/`	slash	$\backslash$	`\backslash`	
$[$	`[ \,`	left [square] bracket	$]$	`] \,`	right [square] bracket
$\langle$	`\langle`	left angle bracket	$\rangle$	`\rangle`	right angle bracket
$\lfloor$	`\lfloor`	floor (left)	$\rfloor$	`\rfloor`	floor (right)
$\llcorner$	`\llcorner`		$\lrcorner$	`\lrcorner`	

Рисунок 2 – Команды LaTex для отображения символов

Как правило, группа определяется как часть документа, заключённая между двумя фигурными скобками. Действие многих команд (таких, как, например, команды выбора шрифта) автоматически отменяется по завершению группы, где они были применены. Кроме открывающей и закрывающей фигурных скобок, в качестве ограничителей могут быть использованы команды \begingroup и \endgroup.

Если LaTeX встречает в тексте символ %, он пропускает текст до конца данной строки, также пропускает разрыв строки и все пробелы в начале следующей строки.

Это может быть использовано для записи заметок в файл, которые не будут отображаться в распечатанной версии.

Индивидуальная работа №3

Используя формат Tex, оформите отчеты по индивидуальному заданию 2, включив описание задачи, математические формулы и программный код.