Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ
ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра Математического и компьютерного моделирования

Информационная система учёт внутриофисных расходов

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 451 группы

направление 38.03.05 — Бизнес информатика

механико-математического факультета

Дранищева Анна Лериевна

Научный руководитель
профессор, д.э.н., профессор Л. В. Кальянов

Зав. кафедрой
зав.каф.,д.ф. – м. н. Ю.А. Блинков

Саратов 2017

ВВЕДЕНИЕ

Актуальной задачей в информационном плане на сегодняшний день для предприятий и корпораций всех организационных форм и видов собственности и в любой предметной области является обеспечение надежного управления всем объемом разнородных данных, которые порождаются, хранятся и используются в различных ИС, существующих на предприятии и связанных с информационной поддержкой продукции (услуг) в течении ее жизненного цикла. Разнообразие проблем, решаемых с помощью ИС, привело к появлению разнотипных систем, различающихся принципами построения и заложенными в них правилами обработки информации.

Данная бакалаврская работа должна показать актуальность и значимость информационных технологий.

Современные предприятия и фирмы представляют собой сложные организационные системы, отдельные составляющие которых — основные и оборотные фонды, трудовые и материальные ресурсы и другие — постоянно изменяются и находятся в сложном взаимодействии друг с другом. Функционирование предприятий и организаций различного типа в условиях рыночной экономики поставило новые задачи по совершенствованию управленческой деятельности на основе комплексной автоматизации управления всеми производственными и технологическими процессами, а также трудовыми ресурсами.

В сфере учета внутриофисных расходов также важна автоматизация, так как она обеспечивает экономию времени руководителей и сотрудников, упрощение работы по оформлению документации, уменьшение вероятности ошибок в работе персонала.

Цель бакалаврской работы заключается в создании информационной системы «Учет внутриофисных расходов».

Будет рассмотрена база данных внутриофисных расходов, созданная с

использованием SQLite, в связке с клиентом, написанным в среде разработки PyQT4. Рассматриваемая программа позволит отслеживать расходы бухгалтерии частной фирмы, их дату, предельную сумму и некоторое описание, причем не только по отделам, но и по отдельным сотрудникам. С помощью удобного интерфейса можно легко добавлять, удалять или редактировать записи в таблицах, что значительно упростит работу с информацией пользователю. Также в данной работе будет создано несколько запросов к базе данных, позволяющих получить более наглядную информацию о внутриофисных расходах фирмы.

Рассмотрим работу бухгалтерии частной фирмы. Сотрудники фирмы имеют возможность осуществлять мелкие покупки для нужд фирмы, предоставляя в бухгалтерию товарный чек. Главной задачей является отслеживание внутриофисных расходов фирмы.

Представленная фирма состоит из отделов. Каждый отдел имеет название. В каждом отделе работает определенное количество сотрудников. По каждому сотруднику имеется личная информация. Сотрудники могут осуществлять покупки в соответствии с видами расходов.

Каждый вид расходов имеет название, некоторое описание и предельную сумму средств, которые могут быть потрачены по данному виду расходов в месяц. При каждой покупке сотрудник оформляет документ, где указывает вид расхода, дату, сумму и отдел.

Нужно хранить данные о расходах не только в целом по отделу, но и по отдельным сотрудникам. Нормативы по расходованию средств устанавливаются не в целом, а по каждому отделу за каждый месяц.

Цель бакалаврской работы – приобретение навыков анализа предметной области, проектирования базы данных, ее физической реализации с помощью SQLite и язык программирования Python.

Для достижения поставленных целей в работе необходимо решить следующие задачи:

- проанализировать предметную область,
- осуществить выбор платформы для разработки ИС,
- провести разработку структуры БД,
- спроектировать и создать пользовательский интерфейс,
- запросы (в режиме Конструктора и на языке SQL),
- выполнить текстовое описание всего произведённого процесса.

Бакалаврская работа состоит из введения, трех разделов, заключения и двух приложений.

Введение содержит постановку задачи и описание предметной области.

Первый раздел содержит обзор предметной области и постановку задачи.

Второй раздел описывает выбор средств для разработки информационной системы, в частности язык моделирования сложных систем UML, и технологии разработки баз данных, СУБД SQLite, язык программирования Python и пакет для создания графического интерфейса PyQT4.

В третьем разделе приводится описание создания требуемой информационной системы. В начале строится модель прецедентов для описания функциональных возможностей разрабатываемой системы. Далее разрабатывается база данных системы и пользовательский интерфейс.

В заключении приводятся результаты проделанной работы.

Основное содержание работы

Необходимо рассмотреть работу бухгалтерии фирмы. Сотрудники фирмы имеют возможность осуществлять мелкие покупки для нужд фирмы, предоставляя в бухгалтерию товарный чек. Задачей информационной системы является отслеживание внутриофисных расходов по каждому сотруднику. Фирма состоит из отделов. Каждый отдел имеет название. В каждом отделе работает определенное количество сотрудников. Информация по каждому сотруднику также заносится в базу данных.

Сотрудники могут осуществлять покупки в соответствии с видами расходов. Каждый вид расходов имеет название, некоторое описание и предельную сумму средств, которые могут быть потрачены по данному виду расходов в месяц. При каждой покупке сотрудник оформляет документ, где указывает вид расхода, дату, сумму и отдел.

Порядок выполнения работы:

- Анализ текстового описания предметной области;
- Выбор структур таблиц и их создание;
- Заполнение данных в таблицы;
- Разработка интерфейса пользователя;
- Создание запросов;
- Создание выходных отчетов.

СУБД SQLite

SQLite — компактная встраиваемая реляционная база данных. Исходный код библиотеки передан в общественное достояние.

Слово «встраиваемый» означает, что SQLite не использует парадигму клиент-сервер, то есть движок SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а предоставляет библиотеку, с которой программа компонуется и движок становится составной частью программы. Таким образом, в качестве протокола обмена

используются вызовы функций (API) библиотеки SQLite. Такой подход уменьшает накладные расходы, время отклика и упрощает программу. SQLite хранит всю базу данных (включая определения, таблицы, индексы и данные) в единственном стандартном файле на том компьютере, на котором исполняется программа. Простота реализации достигается за счёт того, что перед началом исполнения транзакции записи весь файл, хранящий базу данных, блокируется; ACID-функции достигаются в том числе за счёт создания файла журнала.

Несколько процессов или потоков могут одновременно без каких-либо проблем читать данные из одной базы. Запись в базу можно осуществить только в том случае, если никаких других запросов в данный момент не обслуживается; в противном случае попытка записи оканчивается неудачей, и в программу возвращается код ошибки. Другим вариантом развития событий является автоматическое повторение попыток записи в течение заданного интервала времени.

В комплекте поставки идёт также функциональная клиентская часть в виде исполняемого файла sqlite3, с помощью которого демонстрируется реализация функций основной библиотеки. Клиентская часть работает из командной строки, позволяет обращаться к файлу БД на основе типовых функций ОС.

Благодаря архитектуре движка возможно использовать SQLite как на встраиваемых системах, так и на выделенных машинах с гигабайтными массивами данных.

Язык программирования Python

Python — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функций.

Python поддерживает несколько парадигм программирования, в том числе структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений и удобные высокоуровневые структуры данных. Код в Питоне организовывается в функции и классы, которые могут объединяться в модули (они в свою очередь могут быть объединены в пакеты).

Python поддерживает динамическую типизацию, то есть тип переменной определяется только во время исполнения. Поэтому вместо «присваивания значения переменной» лучше говорить о «связывании значения с некоторым именем». В Python имеются встроенные типы: булевый, строка, Unicode-строка, целое число произвольной точности, число с плавающей запятой, комплексное число и некоторые другие. Из коллекций в Python встроены: список, кортеж (неизменяемый список), словарь, множество и другие. Все значения являются объектами, в том числе функции, методы, модули, классы.

Добавить новый тип можно либо написав класс (class), либо определив новый тип в модуле расширения (например, написанном на языке C). Система классов поддерживает наследование (одиночное и множественное) и метапрограммирование. Возможно наследование от большинства встроенных типов и типов расширений.

Все объекты делятся на ссылочные и атомарные. К атомарным относятся int, long, complex и некоторые другие. При присваивании атомарных объектов копируется их значение, в то время как для ссылочных копируется только указатель на объект, таким образом, обе переменные после присваивания используют одно и то же значение. Ссылочные объекты бывают изменяемые и неизменяемые. Например, строки и кортежи являются неизменяемыми, а списки, словари и многие другие объекты —

изменяемыми. Кортеж в Питоне является, по сути, неизменяемым списком. Во многих случаях кортежи работают быстрее списков, поэтому если вы не планируете изменять последовательность, то лучше использовать именно их.

Язык обладает чётким и последовательным синтаксисом, продуманной модульностью и масштабируемостью, благодаря чему исходный код написанных на Питоне программ легко читаем. При передаче аргументов в функции Питон использует вызов по соиспользованию (call-by-sharing).

В программах на Питоне широко используются итераторы. Итератор — объект, абстрагирующий за единым интерфейсом доступ к элементам коллекции. Итератор иногда также называют курсором, особенно если речь идет о базе данных. В Обероне он называется также бегунок и представлен как тип данных. В простейшем случае итератором в низкоуровневых языках является указатель.

Использование итераторов в обобщённом программировании позволяет реализовать универсальные алгоритмы работы с контейнерами Цикл for может работать как с последовательностью, так и с итератором. Все коллекции, как правило, предоставляют итератор. Объекты определённого пользователем класса тоже могут быть итераторами. Подробнее об итераторах можно узнать в разделе о функциональном программировании. Модуль itertools стандартной библиотеки содержит много полезных функций для работы с итераторами.

Одной из интересных возможностей языка являются генераторы — функции, сохраняющие внутреннее состояние: значения локальных переменных и текущую инструкцию (см. также: сопрограммы). Генераторы могут использоваться как итераторы для структур данных и для ленивых вычислений.

При вызове генератора функция немедленно возвращает объект-итератор, который хранит текущую точку исполнения и состояние локальных

переменных функции. При запросе следующего значения (посредством метода next(),неявно вызываемого в цикле for) генератор продолжает исполнение функции от предыдущей точки останова до следующего оператора yield или return. В Python 2.4 появились генераторные выражения — выражения, дающие в результате генератор. Генераторные выражения позволяют сэкономить память там, где иначе требовалось бы использовать список с промежуточными результатами.

С Питоном поставляется библиотека tkinter на основе Tcl/Tk для создания кроссплатформенных программ с графическим интерфейсом.

Существуют расширения, позволяющие использовать все основные библиотеки графических интерфейсов — wxPython, основанное на библиотеке
wxWidgets, PyGTK для Gtk, PyQt и PySide для Qt и другие. Некоторые из них также предоставляют широкие возможности по работе с базами данных, графикой и сетями, используя все возможности библиотеки, на которой основаны.

Для создания игр и приложений, требующих нестандартного интерфейса, можно использовать библиотеку Pygame. Она также предоставляет обширные средства работы с мультимедиа: с её помощью можно управлять звуком и изображениями, воспроизводить видео. Предоставляемое pygame аппаратное ускорение графики OpenGL имеет более высокоуровневый интерфейс по сравнению с PyOpenGL, копирующей семантику C-библиотеки для OpenGL. Есть также PyOgre, обеспечивающая привязку к Ogre — высокоуровневой объектноориентированной библиотеке 3D-графики. Кроме того, существует библиотека pythonOCC обеспечивающая привязку к среде 3D-моделирования и симуляции OpenCascade.

Для работы с растровой графикой используется библиотека Python Imaging Library.

Существуют модули, позволяющие контролировать типы параметров функций на этапе исполнения, например, typecheck или method signature checking decorators. Необязательная декларация типов для параметров функции добавлена в Python 3, интерпретатор при этом не проверяет типы, а только добавляет соответствующую информацию к метаданным функции для последующего использования этой информации модулями расширений.

ЗАКЛЮЧЕНИЕ

В ходе бакалаврской работы была построена модель прецедентов рассматриваемой предметной области, позволившая выявить список потенциальных пользователей разрабатываемой информационной системы, а также ее функциональные возможности. Построенная модель прецедентов приведена в третьем разделе данной работы.

На основании проведенного анализа разработана база данных, позволяющая удовлетворить информационные потребности выявленных пользователей. Процесс создания и описание полученной базы данных, также приведен в третьем разделе.

Был разработан набор экранных форм для работы с базой данных с использованием современных перспективных технологий. Программный код и вид окон приведен в приложении. Разработанная информационная система является учебным проектом, но при соответствующей доработке может быть внедрена в конкретной организации. В ходе эксплуатации возможно внесение изменений с учетом пожеланий конечных пользователей.