СОДЕРЖАНИЕ

|  |  |
| --- | --- |
| Задание | 2 |
| Выбор средств для решения задания | 3 |
| Обоснование выбора средств решения задачи | 4 |
| Алгоритм | 5 |
| Тестовые примеры | 8 |
| Код | 9 |
| ПРИЛОЖЕНИЕ 1 | 11 |
| ПРИЛОЖЕНИЕ 2 | 12 |

Задание

Воспроизведение искусственной человеческой речи из введенного текста.

Входные данные: текст на русском языке.

Выходные данные: искусственная человеческая речь.

Выбор средств для решения задачи

Для решения задачи был выбран язык высокого уровня Python(см. ПРИЛОЖЕНИЕ 1) со средой разработки IDLE и использованием библиотеки pyttsx (см. ПРИЛОЖЕНИЕ 1).

Обоснование выбора средств решения задачи

Языком программирования был выбран **Python** (см. ПРИЛОЖЕНИЕ 1), потому что этот язык ориентирован на повышение производительности разработчика и читаемости кода. Это очень мощный язык, у него много «фанатов», что предоставляет возможность для консультирования и быстрой помощи в случае возникновения трудностей.

Библиотекой была выбрана **pyttsx** (см. ПРИЛОЖЕНИЕ 1), так как она очень легка в использовании, имеет хороший словарь, поддерживает выбор и настройку голоса.

Алгоритм

Все прототипы функций взяты с <http://pyttsx.readthedocs.org/en/v1.1/index.html> и <http://docs.python.org/2.7> .

Вся справочная информация взята с сайта <http://pyttsx.readthedocs.org/en/v1.1/index.html> , <http://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B5%D0%B7_%D1%80%D0%B5%D1%87%D0%B8>

1. Создание движка

Мы создаем не сам движок, а ссылку, которая будет указывать на драйвер синтеза речи (тестирование производилось на драйвере sapi5, для Windows 7)

Прототип функции создания движка:

pyttsx.**init**([driverName : string, debug : bool]) → pyttsx.Engine

Пример создания движка:

|  |
| --- |
| engine = pyttsx.init() |

1. Настройка голоса

Для созданного движка (ссылки на экземпляр движка) необходимо произвести настройки голоса. Можно изменить громкость звука, скорость воспроизведения речи, выбрать голос (выбрать диктора)

Прототип функции для настройки голоса:

**setProperty**(name, value) → None

Пример:

|  |
| --- |
| rate = engine.getProperty('rate') *# получаем скорость воспроизведения*  engine.setProperty('rate', rate - 20) *# слишком быстро, уменьшаем*  engine.setProperty('voice', Alena) *# используем голос Алены* |

1. Воспроизведение синтезированной речи

После того, как голосовой движок(см. ПРИЛОЖЕНИЕ 2) преобразовал текст в речь, необходимо вывести звучание речи.

3.1. Выбор режима ввода текста

Входными данными является текст. Возможны два варианта ввода текста.

3.1.1. Считывание с файла

Пользователь вводит текст в файл с названием “input.txt” и сохраняет его. Программа считывает текст из файла построчно в список, чтобы потом построчно его воспроизвести.

Прототип функции открытия и считывания файла соответственно:

**open**(name[, mode[, buffering]])

file.**readlines**()

Пример:

|  |
| --- |
| text = open('input.txt', 'r+').readlines() *# считываем построчно в файл* |

3.1.2. Считывание с консоли

Пользователь вводит текст в консоль. Программа считывает введенный текст в строку, преобразуя его в Юникод (стандарт кодирования символов, подробней <http://ru.wikipedia.org/wiki/%D0%AE%D0%BD%D0%B8%D0%BA%D0%BE%D0%B4>).

Прототип функции, считывающей входные данные с консоли:

**raw\_input**([prompt])

3.2. Воспроизведение речи

Подаем в функцию текст, получаем озвученную искусственную речь.

Прототип функции воспроизведения речи:

**say**(text : unicode, name : string) → None

**Пример:**

|  |
| --- |
| **for** line **in** text: *# построчно воспроизводим*              engine.say(line.decode('cp1251')) |

3.3. Окончание воспроизведения речи

Для безопасного окончания воспроизведения речи используется функция, которая отслеживает, что очередь пуста, следовательно, синтезирование окончено.

Прототип функции:

**runAndWait**() → None

Тестовые примеры

Данные примеры использовались при тестировании программы. Предлагаю поверить в работоспособность этих примеров, в качестве проверки можно запустить программу и ввести данные тесты.

Пример №1

Стандартное «Hello, world!»

Входные данные: Привет, мир!

Пример №2

Текст из отрывка стихотворения А.С. Пушкина, «Зимний вечер».

Входные данные: Буря мглою небо кроет,

Вихри снежные крутя;

То, как зверь, она завоет,

То заплачет, как дитя,

То по кровле обветшалой

Вдруг соломой зашумит,

То, как путник запоздалый,

К нам в окошко застучит.

Пример №3

Бессмысленный текст, «пример на звучание».

Входные данные: Ауреки рраварршгригш фвф.

Код

|  |
| --- |
| *# -\*- coding: cp1251 -\*-*  **import** pyttsx *# портируем модуль text to speech*    *# Выбор голоса*  Alena = "HKEY\_LOCAL\_MACHINE**\S**OFTWARE**\M**icrosoft**\S**peech**\V**oices**\T**okenEnums**\I**nfovox Desktop v2.2**\A**lyona22k"    engine = pyttsx.init() *# создаем двигатель с стандартным драйвером (текстировал на 'sapi5')*    *# Настройка голоса*  rate = engine.getProperty('rate') *# получаем скорость воспроизведения*  engine.setProperty('rate', rate - 20) *# слишком быстро, уменьшаем*  engine.setProperty('voice', Alena) *# используем голос Алены*    stop = False  **while** stop == False: *# защита от случайного нажатия*      choose = (raw\_input("Ввод из файла(F)/консоли(C): ")).decode('cp1251')  **if** choose == 'F': *# если выбрано считывание с файла*          text = open('input.txt', 'r+').readlines() *# считываем построчно в файл*  **for** line **in** text: *# построчно воспроизводим*              engine.say(line.decode('cp1251'))          engine.runAndWait()          stop = True  **elif** choose == 'C': *# если выбрано считывание с консоли*          text = (raw\_input("Входные данные:  ")).decode('cp1251')          engine.say(text) *# воспроизводим полученный с консоли текст*          engine.runAndWait()          stop = True |

ПРИЛОЖЕНИЕ 1

Коротко о выбранных средствах:

**Python** ([англ.](http://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *python* — [питон](http://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D1%82%D0%BE%D0%BD); в русском языке распространено название *пито́н*)—[высокоуровневый язык программирования](http://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. [Синтаксис](http://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) ядра Python минималистичен. В то же время [стандартная библиотека](http://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%BD%D0%B0%D1%8F_%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_Python) включает большой объём полезных функций.

Python поддерживает несколько [парадигм программирования](http://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%B8%D0%B3%D0%BC%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), в том числе [структурное](http://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [объектно-ориентированное](http://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [функциональное](http://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5),[императивное](http://ru.wikipedia.org/wiki/%D0%98%D0%BC%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) и [аспектно-ориентированное](http://ru.wikipedia.org/wiki/%D0%90%D1%81%D0%BF%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5). Основные архитектурные черты — [динамическая типизация](http://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F), [автоматическое управление памятью](http://ru.wikipedia.org/wiki/%D0%A1%D0%B1%D0%BE%D1%80%D0%BA%D0%B0_%D0%BC%D1%83%D1%81%D0%BE%D1%80%D0%B0), полная [интроспекция](http://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D1%80%D0%BE%D1%81%D0%BF%D0%B5%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), механизм [обработки исключений](http://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D0%B8%D1%81%D0%BA%D0%BB%D1%8E%D1%87%D0%B5%D0%BD%D0%B8%D0%B9), поддержка [многопоточных вычислений](http://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B3%D0%BE%D0%BF%D0%BE%D1%82%D0%BE%D1%87%D0%BD%D0%BE%D1%81%D1%82%D1%8C) и удобные высокоуровневые [структуры данных](http://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85). Код в Питоне организовывается в функции и [классы](http://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), которые могут объединяться в [модули](http://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D1%8C_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) (которые в свою очередь могут быть объединены в пакеты).

(Википедия, <http://ru.wikipedia.org/wiki/Python> )

**pyttsx** – библиотека для синтеза речи из печатного текста.

Поддерживает следующие драйверы:

* *SAPI5* на Windows XP и выше
* *NSSpeechSynthesizer* на Mac OS X 10.5 (Leopard) и 10.6 (Snow Leopard)
* *espeak* на Ubuntu Desktop Edition 8.10 (Intrepid), 9.04 (Jaunty), и 9.10 (Karmic)

(<http://pyttsx.readthedocs.org/en/v1.1/index.html>)

**IDLE** — это [интегрированная среда разработки](http://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) на языке [Python](http://ru.wikipedia.org/wiki/Python), созданная с помощью библиотеки [Tkinter](http://ru.wikipedia.org/wiki/Tkinter). Поставляется вместе с Python.

(Википедия, <http://ru.wikipedia.org/wiki/IDLE> )

ПРИЛОЖЕНИЕ 2

Синтез речи.

*Синтез речи* – это все, что связано с искусственным производством человеческой речи.

*Синтезатор речи* — структура способная переводить текст/образы в речь, в программном обеспечении или аппаратных средствах.

*Голосовой движок* — непосредственно система/ядро преобразования текста/команд в речь, это также может существовать независимо от компьютера.

Применение синтеза речи:

* [информационно-справочные системы](http://ru.wikipedia.org/w/index.php?title=%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%BE-%D1%81%D0%BF%D1%80%D0%B0%D0%B2%D0%BE%D1%87%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0&action=edit&redlink=1)
* помощь слепым и немым
* объявления об отправлении поездов и тому подобное
* выдачи информации о технологических процессах: в военной и авиакосмической технике, в робототехнике, в акустическом диалоге человека с компьютером.
* [звуковой эффект](http://ru.wikipedia.org/wiki/%D0%97%D0%B2%D1%83%D0%BA%D0%BE%D0%B2%D0%BE%D0%B9_%D1%8D%D1%84%D1%84%D0%B5%D0%BA%D1%82) нередко используется в создании [электронной музыки](http://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BC%D1%83%D0%B7%D1%8B%D0%BA%D0%B0).

(Википедия, <http://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B5%D0%B7_%D1%80%D0%B5%D1%87%D0%B8>)