Постановка задачи

Разработать инновационный стековый язык программирования ITER, продумать его концепцию. Реализовать компилятор этого языка в ассемблер.

Используемые алгоритмы и технологии

* Лексический анализ с помощью rply
* Создание консользоного интерфейса с помощью click
* Регулярные выражения
* NASM
* Использование Сишных библиотек при линковке с gcc
* Создание компилятора, рекурсивная обработка токенов
* Архитектура x86 приложения на Linux

Руководство для программиста

Язык программирования основан на идее обратной польской нотации. Он позволяет писать консольные приложения, которые используют исключительно 32-х битные знаковые целые числа. Почти все токены работают по принципу push/pop на реальном программной стеке. Язык способен решать любую задачу в заданных ограничениях с достаточно адекватным и идейным кодом. Для него разрабатывается библиотека алгоритмов algolymp, а также в него добавляются системные функции. Разработчик очень старается сохранять обратную совместимость.

Компилятор написан на языке Python и использует мною написанные ассемблерные макросы, таким образом примерное распределение кода по языкам составляет: Python: 75%, Assembler: 25%. Первым делом ваш код обрабатывается лексером и преобразуется в массив токенов, который затем поступает на обработку в главный класс компилятора. Он последовательно выполняет инклюды модулей, проверку синтаксиса, установку констант директивами %assign, выделение памяти для переменных и массивов в секторе .bss, главную рекурсивную обработку токенов, сборку и линковку выходного ассемблерного файла.

Из библиотек gcc используются функции ввода вывода, аллокатор, а также стандартные функции работы с памятью. В итоговом ассемблерном коде регистр ebp отвечает за изначальное положение стека, регистр esi отвечает виртуальный стек, необходимый для обработки сункций (не путать с функциями), остальные регистры не имеют постоянного назначения, например eax и ebx часто используются для бинарных операций, edx для умножения по модулю, ecx для сдвигов.

Язык демонстрирует хорошую скорость и красивый синтаксис. Его концептуальные особенности позволяют писать альтернативные компиляторы с приличными оптимизациями. Написанных мною компилятор позволяет с легкостью расширять его функционал. Было бы достаточно интересно использовать его для обучения детей как их первый язык программирования, поскольку он способен развить нестандартное мышление.

Руководство для пользователя

Загрузите актуальную версию компилятора с github [репозитория](https://github.com/Gornak40/iter40). Поставьте стандартное питоновское виртуальное окружение и библиотеки из файла зависимостей. Установите дистрибутив Linux и вставьте процессор, поддерживающий набор команд x86. Установите пакеты gcc, gcc-multilibи nasm.Изучите документацию языка, проконсультируйтесь с его [разработчиком](http://t.me/gornak40) или доверенным лицом, чтобы морально и идейно настроиться. Создайте файл с расширением .iter, напишите в нем вашу программу. Вы можете пользоваться компилятором iter.py как обычным консольным приложением, доступные опции будут отображены по команде ./iter.py —help. Чтобы просто скомпилировать программу, выполните ./iter.py source.iter, где source — название вашего файла с программой. Для запуска исполняемого файла выполните ./main.

Литература

* А. В. Стояров; Программирование на языке ассемблера NASM для ОС Linux. - 2011
* <https://click.palletsprojects.com/en/8.1.x/>
* <https://rply.readthedocs.io/en/latest/>
* <https://ru.wikipedia.org/wiki/Обратная_польская_запись>