ЧАСТНА ПРОФЕСИОНАЛНА ГИМНАЗИЯ ЗА ДИГИТАЛНИ НАУКИ
„СОФТУНИ БУДИТЕЛ“, гр. София

ДИПЛОМЕН ПРОЕКТ

на Виктор Веселинов Петров

ученик/ученичка от XII”A” клас

професия- код: **481030**, **“Приложен програмист”**

специалност- код: **4810301**, **“Приложно програмиране”**

Тема: Език за програмиране “VLang”

Ръководител-консултант: Кирил Поповски

Сесия: май-юни 2024г.

Дата: …….

СЪДЪРЖАНИЕ

1. Увод

Добре дошли във VLang: Иновативен Език за Програмиране за Бъдещите Гении. Дипломният проект представя разработка в областта на езиците за програмиране.

В света на бързо развиващите се технологии и компютърни науки, научаването на програмиране става все по-важно. Но за младите ученици и начинаещи програмисти, стъпването в този свят може да изглежда като голямо предизвикателство. Точно тук влиза VLang - иновативен език за програмиране, създаден с цел да бъде лесен за учене, интуитивен за използване и вдъхновяващ за творчество.

Вдъхновен от Нуждите на Бъдещите Програмисти

VLang е резултат от съчетаването на години опит и наблюдения в обучението на програмиране с необходимостта от по-достъпен и лесен за разбиране език. Зад създаването му стои ангажимента да се създаде средство, което да помага на децата и начинаещите програмисти да развият своите умения и да се възползват от вълнуващите възможности, които технологиите предлагат.

Лесен за Учене, Мощен за Употреба

С VLang, ученето на програмиране става лесно и забавно. Езикът е проектиран с удължен фокус върху четимостта и интуитивността на синтаксиса, което го прави идеален за начинаещи. Все пак, това не означава компромис с функционалността. VLang предлага богат набор от функции и възможности, които да разширят творческия потенциал на програмистите и да ги подготвят за по-сложни предизвикателства в бъдеще.

Развиване на Креативността и Логическото Мислене

Важен аспект от обучението в програмирането е развитието на креативността и логическото мислене. VLang е проектиран така, че да насърчава този аспект от ученето, като предоставя прости и елегантни решения за сложни проблеми. Това не само улеснява процеса на учене, но и стимулира учениците да мислят извън обичайните рамки и да създават иновативни програмни решения.

Общност и Ресурси за Подкрепа

В сърцето на VLang стои идеята за общност и сътрудничество. Всеки, който се занимава с програмиране на VLang, е част от глобална общност от студенти, учители, професионалисти и ентусиасти, които споделят знания, опит и проекти. Освен това, са на разположение разнообразни обучителни ресурси, уроци и проекти, които да помогнат на учениците да напреднат в своя учебен път.

Завладейте Света на Програмирането с VLang

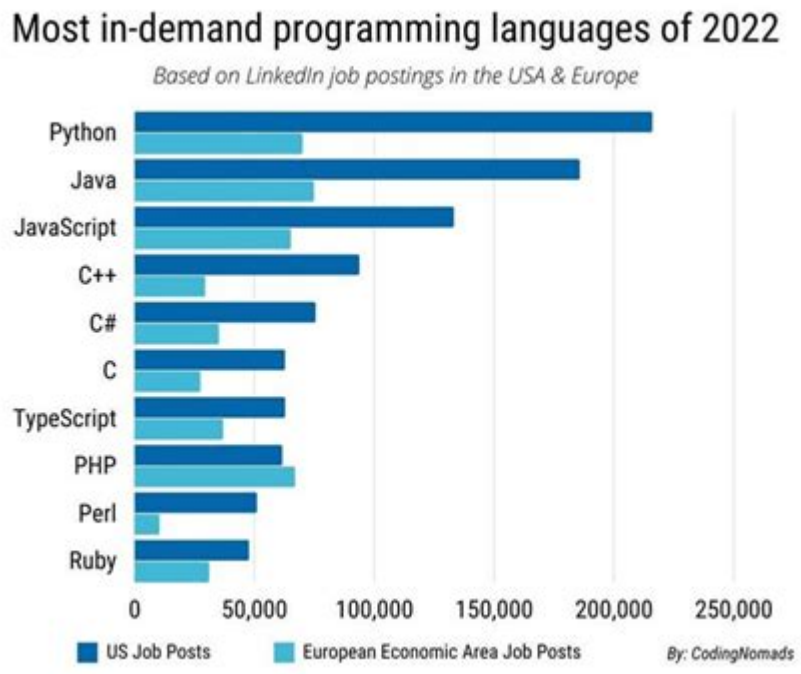
С VLang, бъдещите програмисти могат да разкрият своя потенциал и да се въоръжат със знанията и уменията, необходими за успешна кариера в програмирането. Нека заедно изследваме вълнуващия свят на компютърните науки и да построим бъдещето заедно с VLang - езика, който преобразява ученето във вълнуващо приключение.

2. Изложение

2.1. Глава 1

Ключовата ми мотивация в създаването на проекта беше незадоволеността ми от обучението на млади програмисти– точно в най-ключовия момент, когато тепърва им се създават концепциите за програмния процес.

Започнах да уча програмиране преди едни (вече далечни) 10 години, още в първи клас. Записах се на школа, преподаваща нужните първоначални концепции чрез езика Scratch. Докато сам по себе-си проекта на МТИ (Масачузетския Технологичен Университет) е добре разработен и функционален като инструмент за обучение, в него програмирането се състои от преместване на елементи код с мишка.



Програмирането е пълно с разнообразие от езици, всяко със свои уникални характеристики и специфики. В редица случаи, най-използваните езици за програмиране са тези, които предлагат баланс между ефективност, четимост на кода и разнообразни възможности за разработка.

Python е един от най-популярните езици в света на програмирането поради своята лесна за научаване синтаксис и големия брой библиотеки и инструменти, които

предлага. Синтаксисът на Python е чист и елегантен, като често се използва за учебни цели поради своята яснота и четимост.

JavaScript е незаменим език за уеб разработка. Той позволява динамично променяне на съдържанието на уеб страниците и интерактивност с потребителите. JavaScript се отличава със своя динамичен и гъвкав синтаксис, който позволява създаването на различни приложения, от прости уеб сайтове до сложни уеб базирани приложения.

Java е изключително използван език за програмиране, особено в корпоративната среда. Той се отличава с изключителната си портативност, като код, написан на Java, може да се изпълнява на всяка платформа, която поддържа виртуална машина на Java. Синтаксисът на Java е строг и понякога изисква повече код от други езици, но това осигурява по-голяма яснота и сигурност на приложенията.

C++ е мощен език за програмиране, който се използва широко в различни области като игри, операционни системи и вградени системи. Той се отличава със своята ефективност и възможността за контрол върху хардуера на компютъра. Синтаксисът на C++ е по-сложен в сравнение с някои други езици, като изисква по-голямо внимание при писането на код, но това му позволява да достигне високи нива на производителност.

Първия ми опит с програмиране (чрез клавиатура, като зрял човек) беше с езика Python. Наистина, като за начало беше супер, но имаше определени странни специфики, и дори в някой моменти го смятах за прекалено разглезващ.

Да, очевидно никой няма да кара десет-годишни деца да тръгнат да пишат на C, но все пак няма да е зле да видят и някои по-финни детайли в практика.

Сегашни практики при обучението

В момента, от моя опит и опита на някои мои съученици, процеса на обучение на програмиране е било подобно на това да те учат да плуваш (по много лош начин). Първо ти обясняват теорията– движи едната ръка пред другата и гледай да не се удавиш (това при мен беше общите лекции как работят компютрите). И след това те бутат в дълбокия край на басейна!

Смяната ми от Python към C# (езика, който най-често използваме в училище) беше страшна, странна и нелогична. Имаше много специфики, които научавах за първи път

Една от целите на VLang е смяната към по-сериозни и по-често използвани езици за програмиране (от сорта на C++, Java, C# и т.н.) и по-точно синтаксисът им да бъде по-лесна.

2.2. Глава 2

Семпълност и Четимост на Синтаксиса

VLang трябва да предостави прост и четим синтаксис, който да е достъпен и разбираем за начинаещите програмисти. Програмният код трябва да бъде лесен за въвеждане и разбиране, без излишна сложност или неясноти.

Преход от Лесно Научими Езици към По-Сложни

VLang трябва да предложи постепенен преход от езици като Python и Scratch към по-сложни езици като C++, Java и C#. Това включва улесняване на синтаксиса и въвеждането на концепции, които са общи за по-развитите езици.

Подкрепа за Практически Упражнения и Проекти

Един от ключовите елементи на VLang трябва да бъде подкрепата за практически упражнения и проекти. Това включва предоставяне на широк спектър от примерни проекти, задачи и упражнения, които да помогнат на учениците да приложат своите умения в реални сценарии.

Документация и Обучение

VLang трябва да допълни своята функционалност с обширна документация и обучителни материали. Това включва ръководства за употреба, API документация, видео уроци и онлайн курсове, които да подкрепят учебния процес.

Ангажираност към Общността и Обратна Връзка

За да се осигури постоянното развитие и подобрение на VLang, е от съществено значение да се поддържа активна общност от потребители и програмисти. Това включва възможности за обратна връзка, форуми за обсъждане и сътрудничество върху проекти.

Поддръжка на Различни Платформи и Устройства

VLang трябва да бъде съвместим с различни операционни системи и платформи, включително Windows, macOS и Linux. Това ще улесни достъпа до езика и ще позволи на учениците да учат и да създават програми на различни устройства.

Със своята семпла, но мощна функционалност, VLang се ангажира да бъде инструмент за обучение и вдъхновение за бъдещите програмисти, като им предоставя средствата и ресурсите, необходими за успешното им развитие в света на програмирането.

2.3. Глава 3

Един език за програмиране е комплексен системен инструмент, който позволява на програмистите да създават софтуерни приложения чрез написване на програмен код. Тези езици предоставят разнообразни функционалности и синтаксис, които позволяват на програмистите да изразяват инструкции и алгоритми на машините. Да разгледаме някои от основните функционалности на един типичен език за програмиране:

1. Синтаксис и Граматика

Синтаксисът на езика за програмиране определя правилата за изграждане на валиден програмен код. Това включва правилата за използване на ключови думи, оператори, променливи и структури на данни. Например, в много езици операторът "=" се използва за присвояване на стойност на променлива:

```
int x = 5
```

2. Примитивни Типове Данни

Езиците за програмиране предоставят различни типове данни, които могат да се използват за представяне на информация. Тези типове включват цели числа, дробни числа, символи, низове и булеви стойности. Например, в Python можем да дефинираме променлива от тип цяло число и да извършим математически операции с нея:

```
int num = 10
int result = num * 2
```

3. Оператори и Изрази

Операторите са символи или ключови думи, които извършват операции върху операнди. Те могат да бъдат аритметични, сравнителни, логически и др. Пример за израз с аритметичен оператор в C++:

```
int x = 5;
int y = 3;
int sum = x + y;
```

4. Условни Конструкции

Условните конструкции позволяват на програмата да взема решения в зависимост от различни условия. Най-често използваните са if-else конструкциите. В примера по-долу, ако числото е положително, се извежда съобщение "Числото е положително", в противен случай се извежда "Числото е отрицателно":

```
num = -5
if num > 0:
    print("Числото е положително\n")
else:
    print("Числото е отрицателно\n")
```

5. Цикли

Циклите са конструкции, които позволяват на програмата да изпълнява определени блокове код няколко пъти. Най-често срещаните са for и while циклите. Пример за използване на for цикъл в Java:

```
for (int i = 0; i < 5; i++) {
    System.out.println("Стойността на i е: " + i);
}
```

6. Функции и Методи

Функциите и методите са блокове код, които изпълняват определена задача и могат да бъдат повторно използвани в програмата. Те приемат аргументи и връщат резултат. Пример за дефиниране и извикване на функция в Python:

```
def greet string name:
    return "Hello, " + name + "!"
end greet

message = greet("Alice")
print(message)
```

7. Обработка на Грешки и Изключения

Обработката на грешки и изключения е важна част от програмирането, която позволява на програмата да се справя с непредвидени ситуации. Пример за обработка на грешка в Python:

```
try:
    result = 10 / 0
except ZeroDivisionError:
    print("Деленето на нула не е позволено")
```

Този код представлява език за програмиране, който може да бъде компилиран в междинен език (IL), подобен на C. Ще обясня как работи кодът, включително процесите на парсване (parsing) и лексически анализ (lexing).

8. Лексически анализ

Лексическият анализатор (lexer) сканира входния текст и го разделя на токени (tokens). Това е направено с помощта на регулярни изрази (regex), които съответстват на различни видове токени.

В кода се използват различни регулярни изрази, например:

- `pattern_print` съответства на инструкцията `print`, която извежда стойност на екрана.
- `pattern_variable_assigned` и `pattern_variable_unassigned` съответстват на инструкции за деклариране и инициализиране на променливи.
- `pattern_function` съответства на декларацията на функция.
- `pattern_if`, `pattern_else_if` и `pattern_else` съответстват на конструкции за условен оператор (if-else).
- `pattern_while` съответства на конструкцията за цикъл (while).
- `pattern_return` и `pattern_read` съответстват на инструкции за връщане на стойност и четене от стандартния вход.

След като регулярните изрази се компилират, те се използват за съвпадение със съответните токени в текста.

9. Парсване:

Парсването се извършва върху токените, създадени от лексическия анализатор, за да се определи структурата на програмата. Това позволява на програмата да разбере смисъла на инструкциите и да ги обработи по подходящ начин.

В кода, функцията `validateVlang()` извършва парсването на текста. Тя използва регулярни изрази за да се съпостави с различните видове инструкции и тяхното съдържание. Например, тя разпознава декларации на променливи, дефиниции на функции, инструкции за условен оператор и други.

10. Преобразуване в междинен език :

След като текстът е успешно парснат, програмата генерира код на междинния език (IL), който може да бъде по-лесно преведен в резултатен код. В кода, функцията `parseVLang()` се използва за да се преобразува текстът на програмата в IL формат.

Например, инструкциите за `print` биват преведени в съответните инструкции за извеждане на екрана на C езика, докато декларациите на променливи се превръщат в C декларации със съответните типове данни.

Така, през тези процеси, програмата анализира и преобразува входния текст от езика за програмиране в междинен език, който е по-лесен за обработка и компилация.

11. Сайт на езика

Описание

Този уебсайт е създаден с цел да представи проекта VLang - един нов и иновативен език за програмиране, който се фокусира върху улесняването на процеса на усвояване на програмирането, особено за начинаещи. Сайтът предлага информация за езика, неговите функционалности, инструкции за използване и други свързани ресурси.

Структура на кодовата база на сайта

```
vlang_site
├── app
│   ├── api
│   ├── css
│   └── (default)
├── components
│   ├── ui
│   └── utils
```

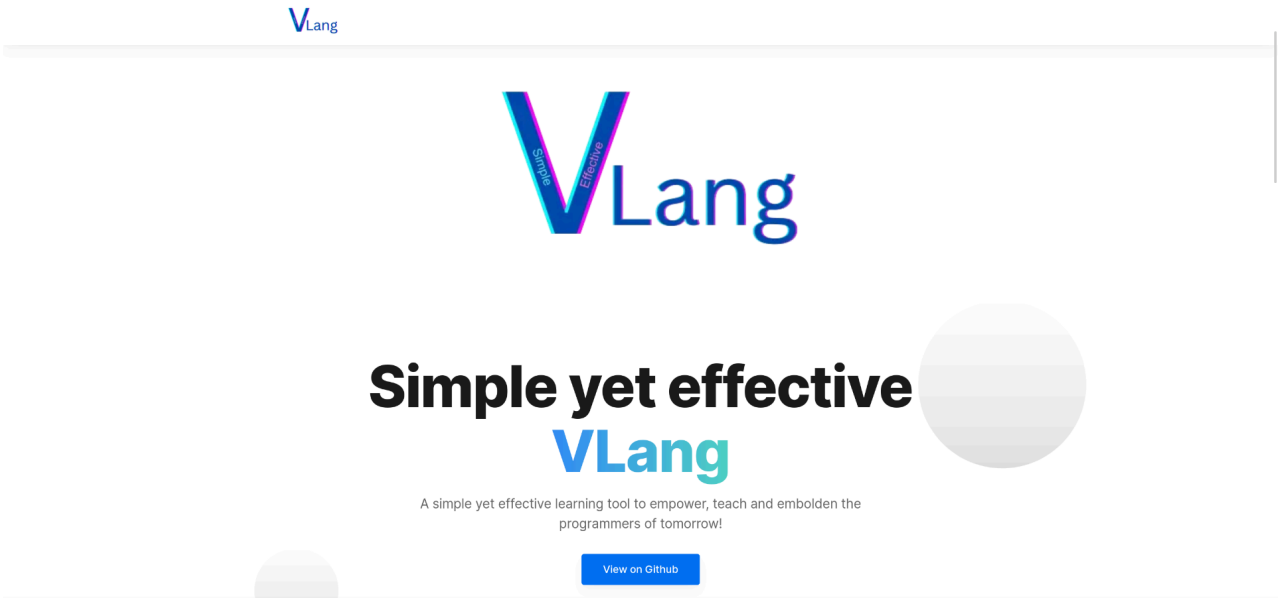
```
└── public
    ├── images
    └── videos
```

Функционалности

Главна страница

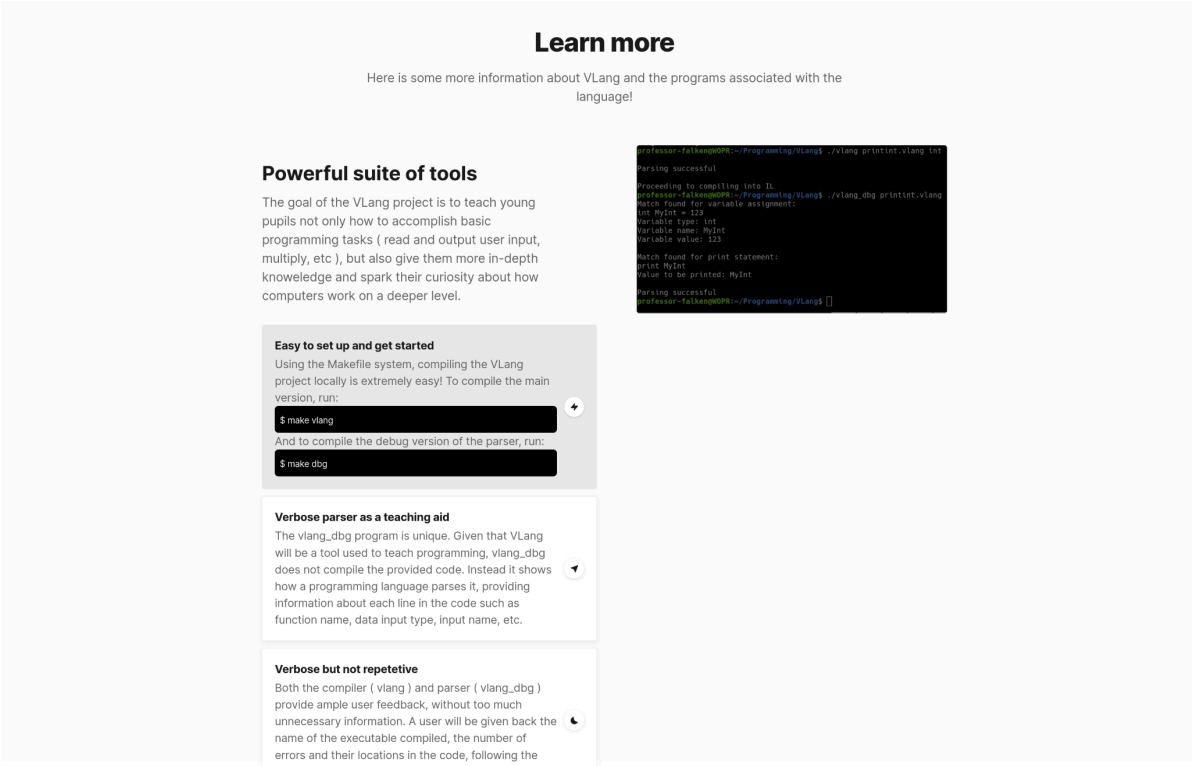
Главната страница на уебсайта представя кратко описание на проекта VLang, заедно с ключови характеристики и предимства на езика. Тук посетителите могат да се запознаят с основната идея зад проекта и да се насочат към други части на уебсайта за допълнителна информация.

Секция "Hero"



Секцията "Hero" е първата нещо, което посетителите ще видят при зареждане на уебсайта. Тя включва привлекателно изображение и кратко послание, което представя същността на проекта VLang.

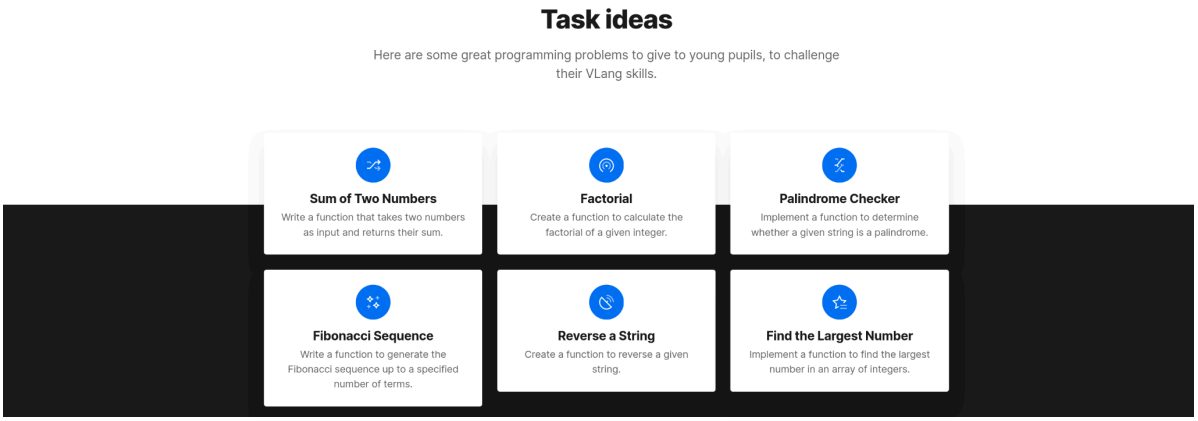
Секция "Features"



Тази секция представя основните функции и възможности на езика VLang. Посетителите могат да се запознаят с това какво прави езика уникален и какви са неговите предимства спрямо други езици за програмиране.

Секция "Features Blocks"

В тази секция са визуално представени няколко различни задачи за ученици. Те са лесни за начинаещи и са чудесен старт в обучението на всички, тръгнали да учат синтаксиса на VLang.



Технологии

TailwindCSS

TailwindCSS е избран като CSS рамка за стилизиране на уебсайта поради своята лекота за употреба и гъвкавост. Той предоставя набор от предварително дефинирани класове, които могат да се използват директно в HTML за стилизиране на елементите.

React/Next.js

React и Next.js са избрани за разработка на клиентската част на уебсайта поради тяхната мощ и гъвкавост. React предоставя лесен начин за създаване на компоненти, които могат да се използват повторно и да се управляват ефективно. Next.js добавя допълнителна функционалност като предварително зареждане на страници и оптимизация за SEO.

Кодова база

page.tsx

Файлът page.tsx е главният компонент за началната страница на уебсайта. Той включва метаданни за страницата и импортира различни компоненти, които се използват за създаване на различни секции на страницата.

FeaturesBlocks

Компонентът FeaturesBlocks е отговорен за показването на различни характеристики и функционалности на езика VLang. Той използва TailwindCSS за стилизиране на елементите и включва различни "блокове" с информация и инструкции за използване.

Node.js модули

Уебсайтът използва множество модули за Node.js, които помагат за създаването и функционирането му. Някои от тези модули включват:

- @tailwindcss: CSS рамка за стилизиране на уебсайта.
- react и react-dom: Библиотеки за разработка на клиентска част с React.
- next: Фреймуърк за създаване на уеб приложения с React.
- typescript: Език за програмиране, който добавя типова проверка към JavaScript.

Тези модули се използват за създаването на различни компоненти и функционалности на уебсайта.

2.4. Глава 4

Описание

"vlang" е компилатор за програмен език, който използва синтаксис, приличащ на езика за програмиране Python, за да генерира междинен код. Този междинен код автоматично се компилира в изпълним файл. Програмата "vlang" има за цел да улесни процеса на разработка на софтуер, като предоставя език, който е по-лесен за усвояване от начинаещи програмисти.

Инсталация

За да използвате програмата "vlang", трябва да я свалите и компилирате от изходния код, който е достъпен на GitHub страницата на проекта. След като сте свалили изходния код, изпълнете следната команда в терминала, за да компилирате програмата:

make vlang
или
make dbg

Това ще генерира изпълним файл с име "vlang" или “vlang_dbg”. “vlang” файлът може да се използва за компилиране на .vlang код в машинен код. “vlang_dbg” файлът може да се използва за да се види задкулисно как работи лексерът на езикът.

Употреба

За да използвате програмата "vlang", изпълнете следната команда в терминала:

./vlang code.vlang output_executable

където:

- code.vlang е файлът с програмния код на "vlang", който искате да компилирате.
- output_executable е името на изходния изпълним файл, към който ще се компилира програмният код.

Грешки

Програмата "vlang" може да върне различни видове грешки в зависимост от проблема в програмния код. Възможните грешки включват, но не са ограничени до:

- Синтактични грешки: Например, некоректно използване на ключови думи или неправилно оформени оператори.
- Логически грешки: Например, некоректни проверки или неправилни изчисления.
- Грешки при компилацията: Например, проблеми при компилацията на междинния код в изпълним файл.

В приложената снимка е демонстрирано хващането на синтактична грешка.

Примерния код в “wrong_program.vlang” се състои от:

string test = "Hello"

pront test

И докато да, променливата е правилно създадена, в “print” изразът има правописна грешка. Компилаторът хваща това и изкарва съответния ред, на който не може да

се намери правилен код:

No code found on line

pront test

```
professor-falken@WOPR:~/Programming/VLang$ cat wrong_program.vlang
string test = "Hello"
pront test
professor-falken@WOPR:~/Programming/VLang$ ./vlang_dbg wrong_program.vlang
Match found for variable assignment:
string test = "Hello"
Variable type: string
Variable name: test
Variable value: "Hello"

No code found on line
pront test


Your code has an error! Fix it!
Error is on line 0
```

Примерен код на "vlang"

```
string greeting = "Welcome to the OddEven program"
print greeting

int num1
int num2
read num1
read num2

int product = num1 * num2

print product

function OddEven int num:

    if num % 2 = 0:
        print num + " is even"
    elseif num %2 != 0:
        print num + "is odd"
    else:
        print "something's wrong, i can feel it"
        return 0
end OddEven

print OddEven(product)
```

Включване на функционалността на "vlang_dbg"

За да използвате допълнителната функционалност на дебъгера за програмата "vlang", можете да използвате следния формат на команда:

./vlang_dbg complex.vlang

Това ще ви даде подробна информация за всяка стъпка на изпълнението на програмата, включително информация за променливите, прочетени стойности и други подобни.

Това е резултатът, който vlang_dbg програмата би дала след като е пусната с аргумент кодът, даден като примерен (OddEven програмата):

Match found for variable assignment:

string greeting = "Welcome to the OddEven program"

Variable type: string

Variable name: greeting

Variable value: "Welcome to the OddEven program"

Match found for print statement:

print greeting

Value to be printed: greeting

Match found for unassigned variable:

int num1

Variable with unassigned value: 1

Match found for unassigned variable:

int num2

Variable with unassigned value: 2

Match found for read statement:

read num1

Read statement value to be inputted: num1

Match found for read statement:

read num2

Read statement value to be inputted: num2

Match found for variable assignment:

int product = num1 * num2

Variable type: int

Variable name: product

Variable value: num1 * num2

Match found for print statement:

print product

Value to be printed: product

Match found for function declaration:

function OddEven int num:

Function name: OddEven

Function input data type: int

Function input variable name: num

Match found for if statement:

if num % 2 = 0:

If statement check: num % 2 = 0

Match found for print statement:

print num + " is even"

Value to be printed: num + " is even"

Match found for else-if statement:

elseif num %2 != 0:

Match found for print statement:

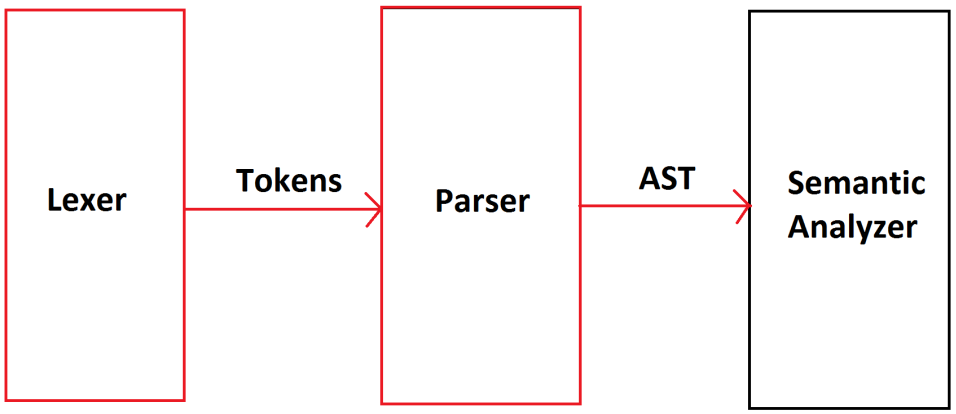
print num + "is odd"

Value to be printed: num + "is odd"

Match found for else:

```
   else:
Match found for print statement:
      print "somenthing's wrong, i can feel it"
Value to be printed: "somenthing's wrong, i can feel it"
Match found for return statement:
      return 0
Return statement value:  0
Match found for function end:
end OddEven
Name of function to be ended: OddEven
Match found for print statement:
print OddEven(product)
Value to be printed: OddEven(product)
Parsing successful
```

Тази програма може да се използва за по-подробно обяснение на парсерът на VLang.



Показва се как се взима под внимание всяка линия код, интерпретира се, разделя се на отделни части (Tokens) и бива тълкувана като част на езика.

Сега сте готови да използвате програмата "vlang" за компилиране на вашия код и разработка на софтуер с нея! Ако имате нужда от допълнителна помощ или информация, не се колебайте да попитате.

3. Заключение

VLang проектът е кулминацията на всичките ми години учене в СофтУни БУДИТЕЛ. Докато, да, е нещо впечатляващо, има още доста допълнителни разработки които се надявам да добавя. Ето кратък списък:

- Да се подкрепя UNICODE
- Да се оптимизира компилатора допълнително
- Кодовата база да се пренапише на C++
- Да се добавят повече примери

4. Информационни източници

Графа Номер 1 (най-търсени езици за програмиране)

https://www.techrepublic.com/article/the-best-programming-languages-to-learn-in-2022/

Всички други снимки и графи използвани в документацията са направени от Виктор Петров

5. Приложения

За употреба на vlang и vlang_dbg програмата са нужни следните програми

- GNU make
- gcc

Версиите им и дали са инсталирани може да се проверят по следния начин:
“make -v”

за GNU make и

“gcc -v”

за gcc.

За сайта е нужен “npm”. Проверява се дали е инсталиран с “npm -v”. Лично препоръчвам програмата “nvm”-- Node Version Selector. След това в директорията на проекта трябва да се пусне командата “npm install” за изтегляне на всички нужни проекти и “npm run dev” за локално стартиране на сайта на порт 3000 за разработчици.

Версиите, използвани в разработката:

gcc 11.4.0

GNU make 4.3

node v22.0.0