asm

```asm
.intel_syntax
.global _start

.section .text

_start:
    call        main..main
    mov         %rdi, 0
    call        exit

main..main:
    push        %rbp
    mov         %rbp, %rsp
    sub         %rsp, 32
    mov         qword ptr [%rbp-8], 3
    lea         %rax, qword ptr [%rbp-8]
    mov         qword ptr [%rbp-16], %rax
    lea         %rax, qword ptr [%rbp-16]
    mov         qword ptr [%rbp-24], %rax
    mov         %rax, qword ptr [%rbp-24]
    mov         %rax, qword ptr [%rax]
    mov         qword ptr [%rbp-32], %rax
    mov         %rdi, qword ptr [%rbp-24]
    mov         %rdi, qword ptr [%rdi]
    mov         %rdi, qword ptr [%rdi]
    mov         %rsi, qword ptr [%rbp-24]
    mov         %rsi, qword ptr [%rsi]
    mov         %rsi, qword ptr [%rsi]
    call        __rush_internal_pow_int
    mov         %rdi, %rax
    mov         %rax, qword ptr [%rbp-32]
    mov         qword ptr [%rax], %rdi
    mov         %rdi, qword ptr [%rbp-24]
    mov         %rdi, qword ptr [%rdi]
    mov         %rdi, qword ptr [%rdi]
    call        exit
main..main.return:
    leave
    ret
```

c

```c
int fib(int n) {
    if (n < 2) {
        return n;
    }
    return fib(n - 1) + fib(n - 2);
}
```

c_sharp

```c_sharp
public static ulong Fib(uint x) {
    if (x == 0) return 0;

    ulong prev = 0;
    ulong next = 1;
    for (int i = 1; i < x; i++)
    {
        ulong sum = prev + next;
        prev = next;
        next = sum;
    }
    return next;
}
```

comment

missing example program

```css
:root {
    --bg-dark: #000;
}

#app.dark {
    background-color: var(--bg-dark);
}
```

dart

```dart
int fib(int n) {
  if (n==0 || n==1) {
    return n;
  }
  var prev=1;
  var current=1;
  for (var i=2; i<n; i++) {
    var next = prev + current;
    prev = current;
    current = next;
  }
  return current;
}

int fibRec(int n) => n==0 || n==1 ? n : fibRec(n-1) + fibRec(n-2);

main() {
  print(fib(11));
  print(fibRec(11));
}
```

diff

```
diff --git a/xtask/src/add_lang.rs b/xtask/src/add_lang.rs
index 990eae60..f535802a 100644
--- a/xtask/src/add_lang.rs
+++ b/xtask/src/add_lang.rs
@@ -13,7 +13,7 @@ use once_cell::sync::Lazy;
 use serde_json::{Map, Value};

 static URL_REGEX: Lazy<Regex> =
-    Lazy::new(|| Regex::new(r"https:\/\/github\.com\/([^\/]*)\/([^\/?#]*)").unwrap());
+    Lazy::new(|| Regex::new(r"https:\/\/(github|gitlab)\.com\/([^\/]*)\/([^\/?#]*)").unwrap());

 pub fn run() -> Result<()> {
     let group = env::args()
@@ -30,10 +30,17 @@ pub fn run() -> Result<()> {
     let rev = get_rev(&url).with_context(|| "unable to fetch latest revision of repository")?;

     let content_url = match URL_REGEX.captures(&url) {
-        Ok(Some(groups)) => Some(format!(
-            "https://raw.githubusercontent.com/{}/{}/{rev}",
-            &groups[1], &groups[2],
-        )),
+        Ok(Some(groups)) => match &groups[1] {
+            "github" => Some(format!(
+                "https://raw.githubusercontent.com/{}/{}/{rev}",
+                &groups[2], &groups[3],
+            )),
+            "gitlab" => Some(format!(
+                "https://gitlab.com/{}/{}/-/raw/{rev}",
+                &groups[2], &groups[3],
+            )),
+            _ => unreachable!("the regex only allows above options"),
+        },
         _ => None,
     };
     let path_in_url = match &path {
```

ebnf

```
Program = { Item } ;

Item               = FunctionDefinition | LetStmt ;
FunctionDefinition = 'fn' , ident , '(' , [ ParameterList ] , ')'
                   , [ '->' , Type ] , Block ;
ParameterList      = Parameter , { ',' , Parameter } , [ ',' ] ;
Parameter          = [ 'mut' ] , ident , ':' , Type ;

Block = '{' , { Statement } , [ Expression ] , '}' ;
Type  = { '*' } , ( ident
                  | '(' , ')' ) ;

Statement    = LetStmt | ReturnStmt | LoopStmt | WhileStmt | ForStmt
             | BreakStmt | ContinueStmt | ExprStmt ;
LetStmt      = 'let' , [ 'mut' ] , ident , [ ':' , Type ] , '='
             , Expression , ';' ;
ReturnStmt   = 'return' , [ Expression ] , ';' ;
LoopStmt     = 'loop' , Block , [ ';' ] ;
WhileStmt    = 'while' , Expression , Block , [ ';' ] ;
ForStmt      = 'for' , ident , '=' , Expression , ';' , Expression
             , ';' , Expression , Block , [ ';' ] ;
BreakStmt    = 'break' , ';' ;
ContinueStmt = 'continue' , ';' ;
ExprStmt     = ExprWithoutBlock , ';'
             | ExprWithBlock , [ ';' ] ;

Expression       = ExprWithoutBlock | ExprWithBlock ;
ExprWithBlock    = Block | IfExpr ;
IfExpr           = 'if' , Expression , Block , [ 'else' , ( IfExpr
                                                          | Block ) ] ;
ExprWithoutBlock = int
                 | float
                 | bool
                 | char
                 | ident
                 | PrefixExpr
                 | InfixExpr
                 | AssignExpr
                 | CallExpr
                 | CastExpr
                 | '(' , Expression , ')' ;
PrefixExpr       = PREFIX_OPERATOR , Expression ;
InfixExpr        = Expression , INFIX_OPERATOR , Expression ;
(* The left hand side can only be an `ident` or a `PrefixExpr` with the `*` operator *)
AssignExpr       = Expression , ASSIGN_OPERATOR , Expression ;
CallExpr         = ident , '(' , [ ArgumentList ] , ')' ;
ArgumentList     = Expression , { ',' , Expression } , [ ',' ] ;
CastExpr         = Expression , 'as' , Type ;

ident = LETTER , { LETTER | DIGIT } ;
int   = DIGIT , { DIGIT | '_' }
      | '0x' , HEX , { HEX | '_' } ;
float = DIGIT , { DIGIT | '_' } , ( '.' , DIGIT , { DIGIT | '_' }
                                   | 'f' ) ;
char  = "'" , ( ASCII_CHAR - '\'
              | '\' , ( ESCAPE_CHAR
                      | "'"
                      | 'x' , 2 * HEX ) ) , "'" ;
bool  = 'true' | 'false' ;

comment = '//' , { CHAR } , ? LF ?
        | '/*' , { CHAR } , '*/' ;

LETTER      = 'A' | 'B' | 'C' | 'D' | 'E' | 'F' | 'G' | 'H' | 'I'
            | 'J' | 'K' | 'L' | 'M' | 'N' | 'O' | 'P' | 'Q' | 'R'
            | 'S' | 'T' | 'U' | 'V' | 'W' | 'X' | 'Y' | 'Z' | 'a'
            | 'b' | 'c' | 'd' | 'e' | 'f' | 'g' | 'h' | 'i' | 'j'
            | 'k' | 'l' | 'm' | 'n' | 'o' | 'p' | 'q' | 'r' | 's'
            | 't' | 'u' | 'v' | 'w' | 'x' | 'y' | 'z' | '_' ;
DIGIT       = '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8'
            | '9' ;
HEX         = DIGIT | 'A' | 'B' | 'C' | 'D' | 'E' | 'F' | 'a'
            | 'b' | 'c' | 'd' | 'e' | 'f' ;
CHAR        = ? any UTF-8 character ? ;
ASCII_CHAR  = ? any ASCII character ? ;
ESCAPE_CHAR = '\' | 'b' | 'n' | 'r' | 't' ;

PREFIX_OPERATOR     = '!' | '-' | '&' | '*' ;
INFIX_OPERATOR      = ARITHMETIC_OPERATOR | RELATIONAL_OPERATOR
                    | BITWISE_OPERATOR | LOGICAL_OPERATOR ;
ARITHMETIC_OPERATOR = '+' | '-' | '*' | '/' | '%' | '**' ;
RELATIONAL_OPERATOR = '==' | '!=' | '<' | '>' | '<=' | '>=' ;
BITWISE_OPERATOR    = '<<' | '>>' | '|' | '&' | '^' ;
LOGICAL_OPERATOR    = '&&' | '||' ;
ASSIGN_OPERATOR     = '=' | '+=' | '-=' | '*=' | '/=' | '%='
                    | '**=' | '<<=' | '>>=' | '|=' | '&=' | '^=' ;
```

ejs

```
<span>
  <% console.log('Hello, World!') -%>
</span>
```

erb

```erb
<div>
  <% names.each do |name| _%>
    <div>
      <%= name -%>
      <%== x %>
      <%|= x %>
      <%== x %>
      <%| end %>
    </div>
    <span>
      <% something() -%>
    </span>
  <%_ end %>
</div>

<%graphql
  fragment HumanFragment on Human {
    name
    homePlanet
  }
%>

<p><%= human.name %> lives on <%= human.home_planet %>.</p>
```

go

```go
import (
    "math/big"
)

func fib(n uint64) *big.Int {
    if n < 2 {
        return big.NewInt(int64(n))
    }
    a, b := big.NewInt(0), big.NewInt(1)
    for n--; n > 0; n-- {
        a.Add(a, b)
        a, b = b, a
    }
    return b
}

func main() {
    regexp.Compile(`[a-fA-F0-9_]\s(.*)$`)
}
```

hexdump

```
0000:   0061 736d 0100 0000  010d 0360 017f 0060   .asm.......`...`
0010:   0000 6001 7e01 7e02  2401 1677 6173 695f   ..`.~.~.$..wasi_
0020:   736e 6170 7368 6f74  5f70 7265 7669 6577   snapshot_preview
0030:   3109 7072 6f63 5f65  7869 7400 0003 0302   1.proc_exit.....
0040:   0102 0503 0100 0007  1302 065f 7374 6172   ...........star
0050:   7400 0106 6d65 6d6f  7279 0200 0801 010a   t...memory......
0060:   2902 0a00 420a 1002  a710 0000 0b1c 0020   )...B..........
0070:   0042 0253 047e 2000  0520 0042 027d 1002   .B.S.~ .. .B.}..
0080:   2000 4201 7d10 027c  0b0b 002a 046e 616d    .B.}..|...*.nam
0090:   6501 1903 000b 5f5f  7761 7369 5f65 7869   e.....__wasi_exi
00a0:   7401 046d 6169 6e02  0366 6962 0208 0201   t..main..fib....
00b0:   0002 0100 016e                            .....n
```

java

```java
class Fibonacci {
    /**
    * O(log(n))
    */
    public static long fib(long n) {
        if (n <= 0)
        return 0;

        long i = (int) (n - 1);
        long a = 1, b = 0, c = 0, d = 1, tmp1,tmp2;

        while (i > 0) {
            if (i % 2 != 0) {
                tmp1 = d * b + c * a;
                tmp2 = d * (b + a) + c * b;
                a = tmp1;
                b = tmp2;
            }

            tmp1 = (long) (Math.pow(c, 2) + Math.pow(d, 2));
            tmp2 = d * (2 * c + d);

            c = tmp1;
            d = tmp2;

            i = i / 2;
        }
        return a + b;
    }
}
```

javascript

```javascript
/**
 * Calculate a number as of the Fibonacci sequence.
 *
 * @example
 * var result = fib(10); // results in 55
 *
 * @param {number} n: index of number to calculate
 */
var fib = (function(cache){
    return cache = cache || {}, function(n){
        if (cache[n]) return cache[n];
        else return cache[n] = n == 0 ? 0 : n < 0 ? -fib(-n)
            : n <= 2 ? 1 : fib(n-2) + fib(n-1);
    };
})();
```

`jsdoc`

missing example program

```json
{
    "key": "value",
    "good": false,
    "age": 42,
    "percentage": 0.3,
    "nothing": null,
    "list": [1, 2, 3],
    "object": {
        "key": "value"
    }
}
```

json5

```
{
    key: "value",
    good: false,
    age: 42,
    percentage: 0.3,
    nothing: null,
    list: [1, 2, 3],
    // NOTE: line comment
    object: {
        "key": /* block comment */ "value",
    },
}
```

jsonc

```jsonc
{
    "key": "value",
    "good": false,
    "age": 42,
    "percentage": 0.3,
    "nothing": null,
    "list": [1, 2, 3],
    // line comment
    "object": {
        "key": /* block comment */ "value"
    }
}
```

```ql
/**
 * @name Information disclosure through postMessage
 * @description Tracks values from an 'authKey' property into a postMessage call with unrestricted origin,
 *              indicating a leak of sensitive information.
 * @kind path-problem
 * @problem.severity warning
 * @tags security
 * @id js/examples/information-disclosure
 */

import javascript
import DataFlow
import DataFlow::PathGraph

/**
 * A dataflow configuration that tracks authentication tokens ("authKey")
 * to a postMessage call with unrestricted target origin.
 *
 * For example:
 * ```
 * win.postMessage(JSON.stringify({
 *  action: 'pause',
 *  auth: {
 *    key: window.state.authKey
 *  }
 * }), '*');
 * ```
 */
class AuthKeyTracking extends DataFlow::Configuration {
  AuthKeyTracking() { this = "AuthKeyTracking" }

  override predicate isSource(Node node) { node.(PropRead).getPropertyName() = "authKey" }

  override predicate isSink(Node node) {
    exists(MethodCallNode call |
      call.getMethodName() = "postMessage" and
      call.getArgument(1).getStringValue() = "*" and // no restriction on target origin
      call.getArgument(0) = node
    )
  }

  override predicate isAdditionalFlowStep(Node pred, Node succ) {
    // Step into objects: x -> { f: x }
    succ.(SourceNode).getAPropertyWrite().getRhs() = pred
    or
    // Step through JSON serialization: x -> JSON.stringify(x)
    // Note: TaintTracking::Configuration includes this step by default, but not DataFlow::Configuration
    exists(CallNode call |
      call = globalVarRef("JSON").getAMethodCall("stringify") and
      pred = call.getArgument(0) and
      succ = call
    )
  }
}

from AuthKeyTracking cfg, PathNode source, PathNode sink
where cfg.hasFlowPath(source, sink)
select sink.getNode(), source, sink, "Message leaks the authKey from $@.", source.getNode(), "here"
```

regex

missing example program

rush

```rush
// Calculates a number in the Fibonacci sequence.
fn main() {
    // fib(10) = 55
    exit(fib(10));
}

fn fib(n: int) -> int {
    if n < 2 {
        n
    } else {
        fib(n - 2) + fib(n - 1)
    }
}
```

```rust
fn fib(n: usize) -> usize {
    if n < 2 {
        n
    } else {
        fib(n - 1) + fib(n - 2)
    }
}

// NOTE: test
fn main() {
    Regex::new(r"[a-fA-F0-9_]\s(.*)$");
    let a = regex!(r"[a-fA-F0-9_]\s(.*)$");
    if regex_is_match!(/* comment */ r"[a-fA-F0-9_]\s(.*)$"i, r"raw text \s[a-f]") {
        return;
    }
}
```

scss

```scss
@use "sass:math";

$font-stack: Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;
}

nav {
  ul {
    margin: 0;
    padding: 0;
    list-style: none;
  }

  li { display: inline-block; }

  a {
    display: block;
    padding: 6px 12px;
    text-decoration: none;
  }
}

@mixin theme($theme: DarkGray) {
  background: $theme;
  box-shadow: 0 0 1px rgba($theme, .25);
  color: #fff;
}

.info {
  @include theme;
}
.alert {
  @include theme($theme: DarkRed);
}
.success {
  @include theme($theme: DarkGreen);
}

article[role="main"] {
  width: math.div(600px, 960px) * 100%;
}
```

toml

```toml
[package]
name = "syntastica"
version = "0.3.0"
authors.workspace = true
edition = "2021"
keywords = ["tree-sitter", "syntect", "highlight", "parsing", "syntax"]
license.workspace = true
repository.workspace = true
description = "Modern and easy syntax highlighting using tree-sitter"
```

tsx

```tsx
interface FooProp {
  name: string;
  X: number;
  Y: number;
}
declare function AnotherComponent(prop: { name: string });
function ComponentFoo(prop: FooProp) {
  return <AnotherComponent name={prop.name} />;
}
const Button = (prop: { value: string }, context: { color: string }) => (
  <button />
);
```

typescript

```typescript
interface User {
  name: string;
  id: number;
}

class UserAccount {
  name: string;
  id: number;

  constructor(name: string, id: number) {
    this.name = name;
    this.id = id;
  }
}

const user: User = new UserAccount("Murphy", 1);
```

verilog

```verilog
module toplevel(clock,reset);
  input clock;
  input reset;

  reg flop1;
  reg flop2;

  always @ (posedge reset or posedge clock)
    if (reset)
      begin
        flop1 <= 0;
        flop2 <= 1;
      end
    else
      begin
        flop1 <= flop2;
        flop2 <= flop1;
      end
endmodule
```

wat

```
(module
  (type (;0;) (func (param i32)))
  (type (;1;) (func))
  (type (;2;) (func (param i64) (result i64)))
  (import "wasi_snapshot_preview1" "proc_exit" (func $__wasi_exit (;0;) (type 0)))
  (func $main (;1;) (type 1)
    i64.const 10
    call $fib
    i32.wrap_i64
    call $__wasi_exit
    unreachable
  )
  (func $fib (;2;) (type 2) (param $n i64) (result i64)
    local.get $n
    i64.const 2
    i64.lt_s
    if (result i64) ;; label = @1
      local.get $n
    else
      local.get $n
      i64.const 2
      i64.sub
      call $fib
      local.get $n
      i64.const 1
      i64.sub
      call $fib
      i64.add
    end
  )
  (memory (;0;) 0)
  (export "_start" (func $main))
  (export "memory" (memory 0))
  (start $main)
)
```