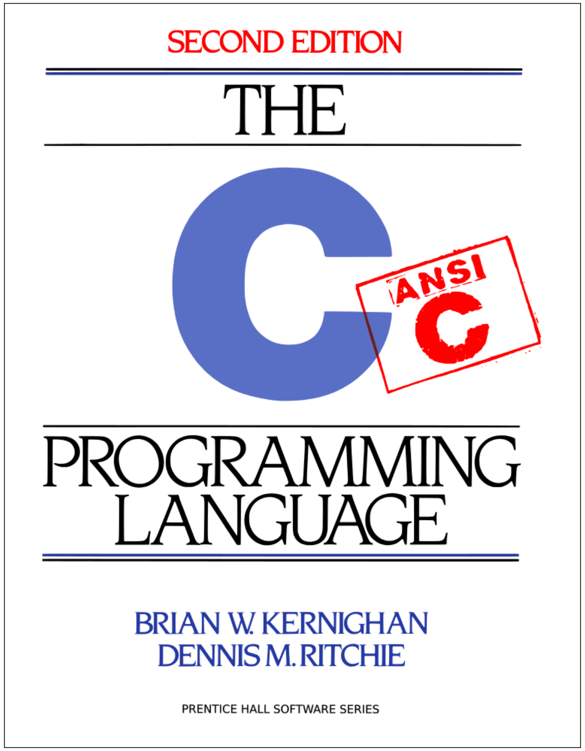
Cπ

Edward Garcia (ewg2115)
Naveen Revanna (nr2443)
Niket Kandya (nk2531)
Sean Yeh (smy2112)

Introduction



SECOND EDITION

THE

C

ANSI C

PROGRAMMING LANGUAGE

BRIAN W. KERNIGHAN
DENNIS M. RITCHIE

PRENTICE HALL SOFTWARE SERIES

Subset of C

ARM®

ARM V6 Assembly

RaspberryPi

Supported Features

Functions:

malloc
free
printf
scanf

Types:

int
char
void
struct
pointer (to anything, unlimited levels)
array

Control/looping:

if
else
while
for
return

Operators:

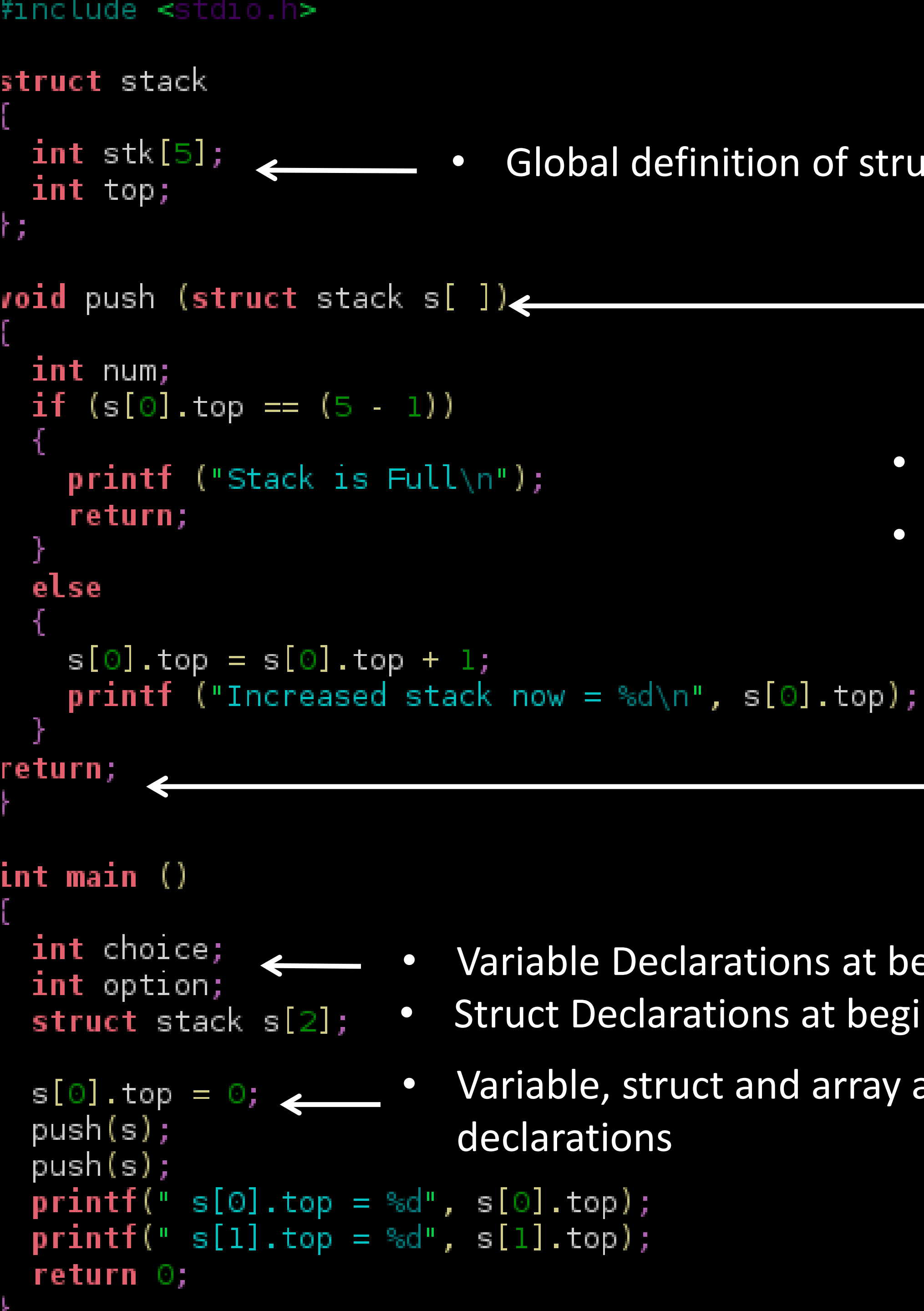
+ - * /
< <= == > >=
&& ||

Most of your favorite features from C...

Unsupported Features

- double, float types
- floating point operations
- short and long integers
- Unsigned, signedintegers
- break, continue
- Enums
- Sizeof()
- Increment, decrement operators.
- do-while and switch statements.
- auto, register, volatile static and extern.
- Multi-file compilation and linkage.
- Preprocessing - no # directives.
- Function pointers.
- Function inlining.
- Static and volatile function.
- Variable function arguments - Ellipsis (...)
- Typecasting

Scoping



```
#include <stdio.h>

struct stack
{
  int stk[5];
  int top;
};

void push (struct stack s[ ])
{
  int num;
  if (s[0].top == (5 - 1))
  {
    printf ("Stack is Full\n");
    return;
  }
  else
  {
    s[0].top = s[0].top + 1;
    printf ("Increased stack now = %d\n", s[0].top);
  }
return;
}

int main ()
{
  int choice;
  int option;
  struct stack s[2];

  s[0].top = 0;
  push(s);
  push(s);
  printf(" s[0].top = %d", s[0].top);
  printf(" s[1].top = %d", s[1].top);
  return 0;
}
```

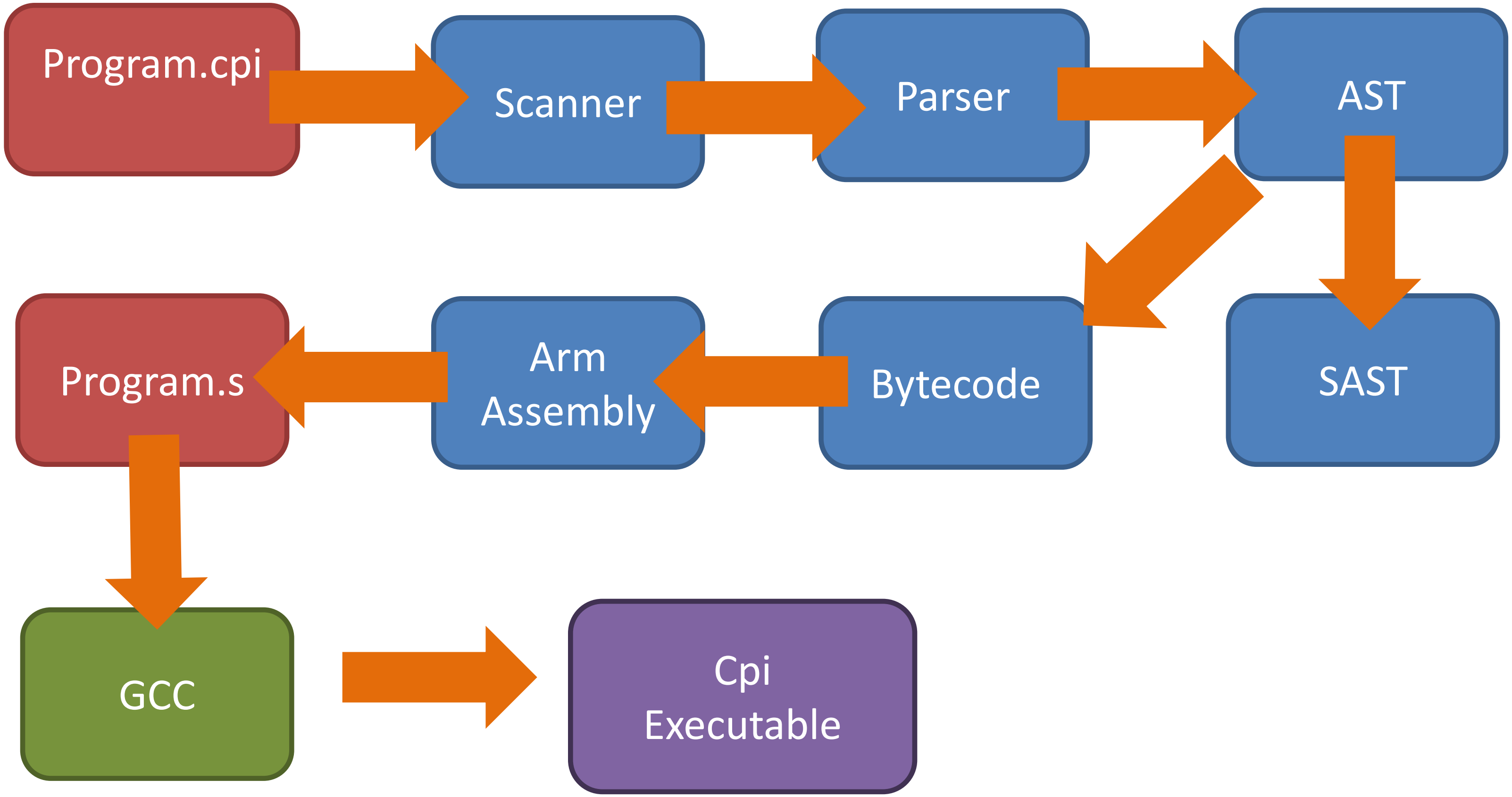
- Global definition of structs

- Variable Scope Limited to function
- Static Scoping

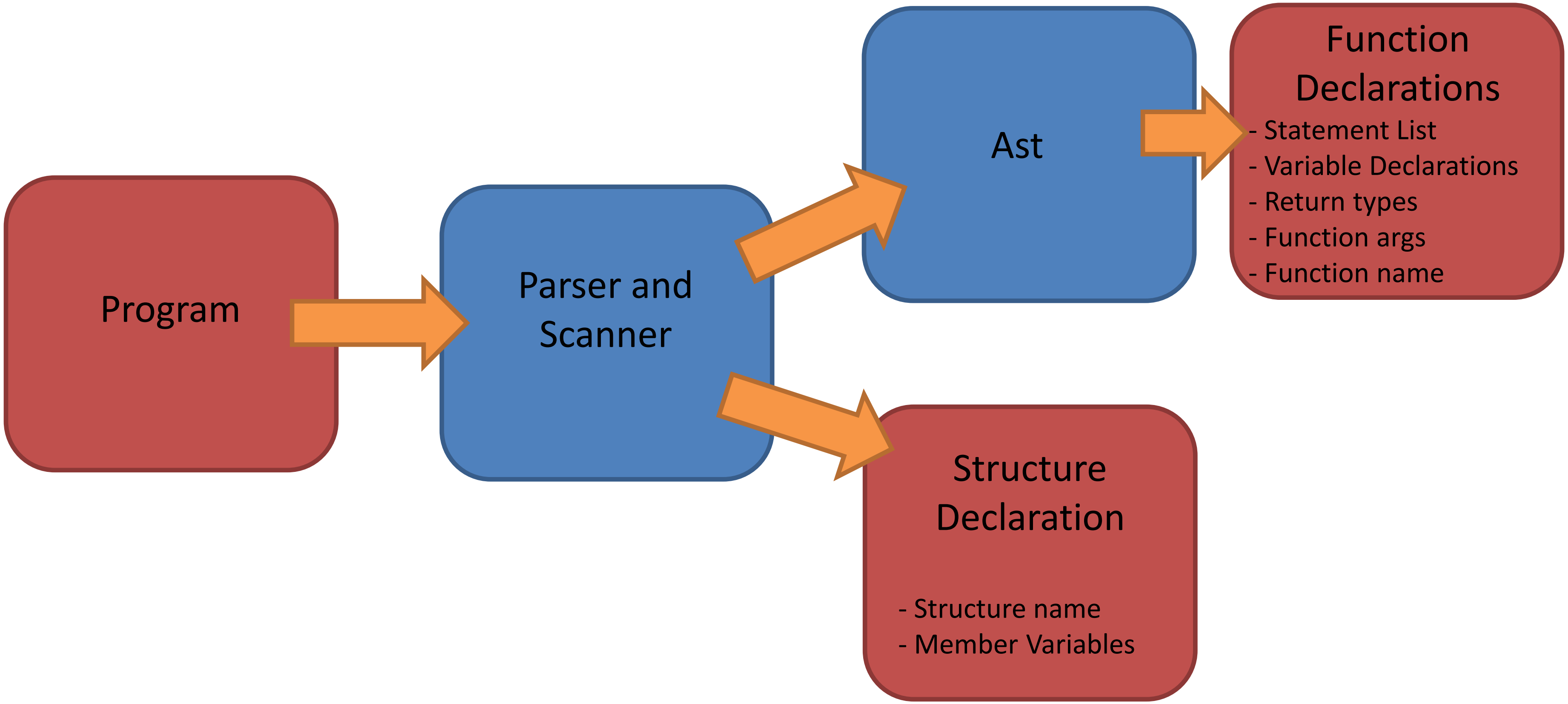
- Variable Declarations at beginning of functions
- Struct Declarations at beginning of functions

- Variable, struct and array assignment following declarations

Architecture



Parser / Ast



Program

Parser and Scanner

Ast

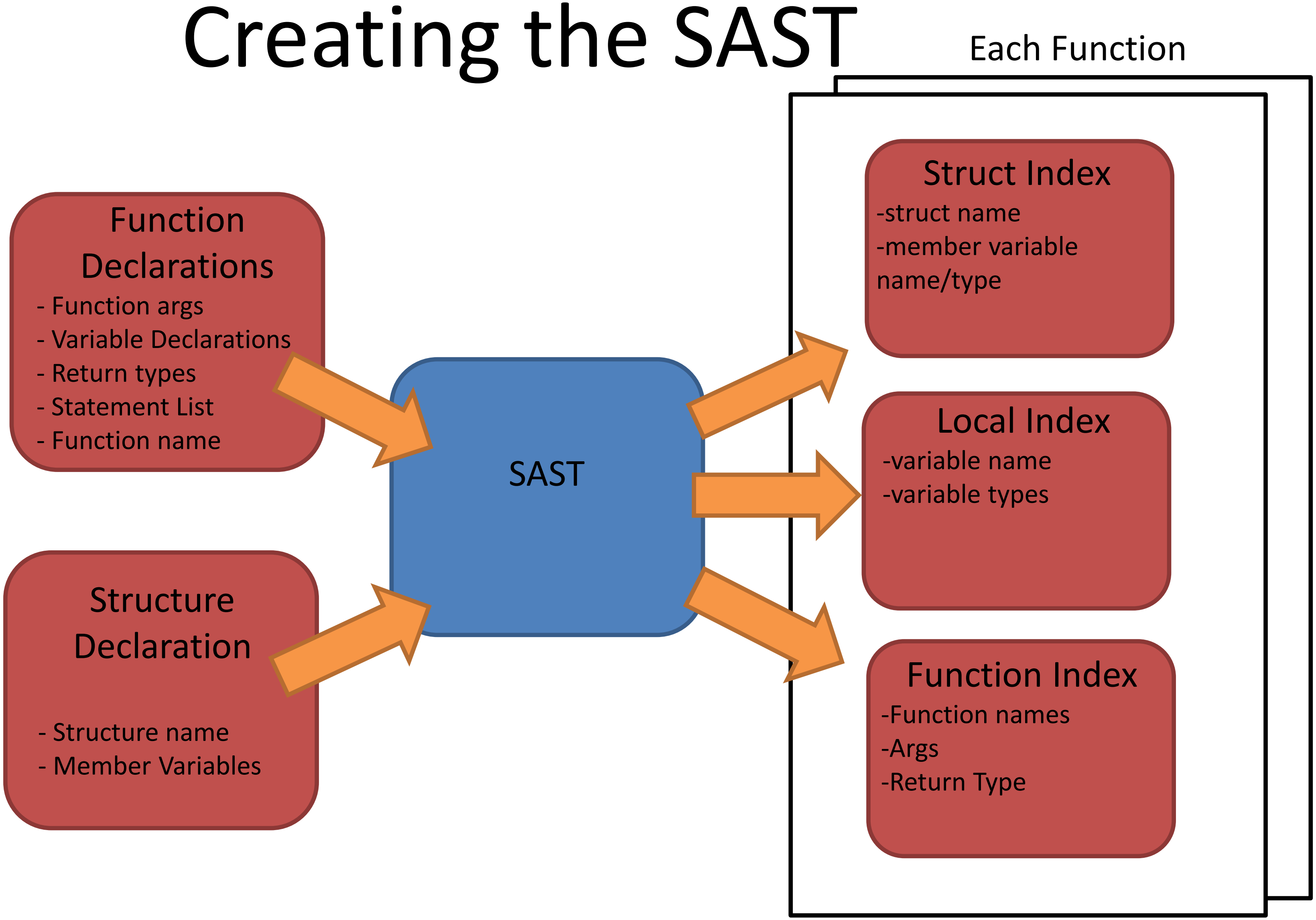
Function Declarations

- Statement List
- Variable Declarations
- Return types
- Function args
- Function name

Structure Declaration

- Structure name
- Member Variables

Creating the SAST



Each Function

Function Declarations
- Function args
- Variable Declarations
- Return types
- Statement List
- Function name

Structure Declaration

- Structure name
- Member Variables

SAST

Struct Index
-struct name
-member variable name/type

Local Index
-variable name
-variable types

Function Index
-Function names
-Args
-Return Type

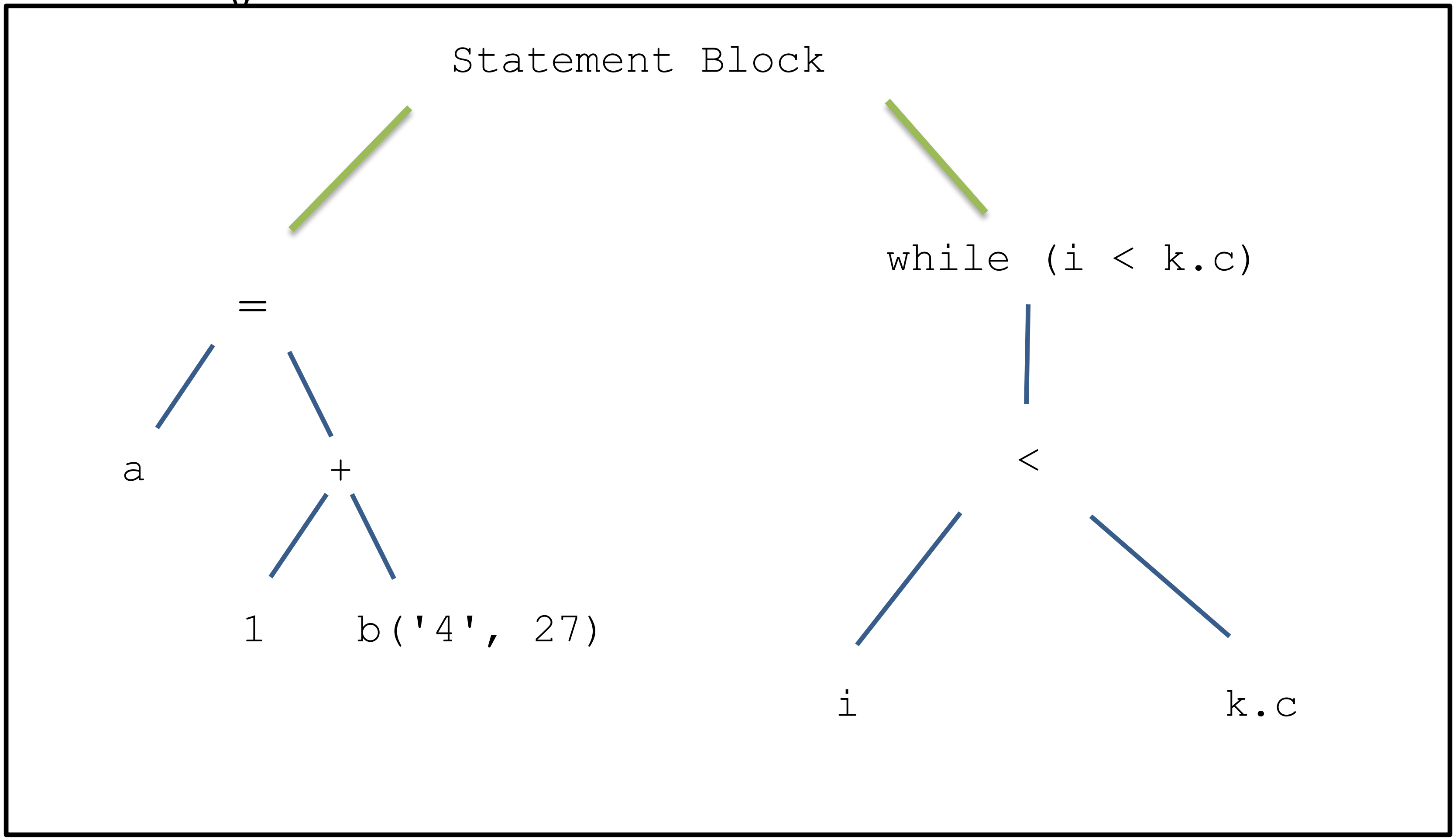
Creating the SAST

Function Index

Struct Index

Local Index

Function Ex()



Statement Block

=

a

+

1

b('4', 27)

while (i < k.c)

<

i

k.c

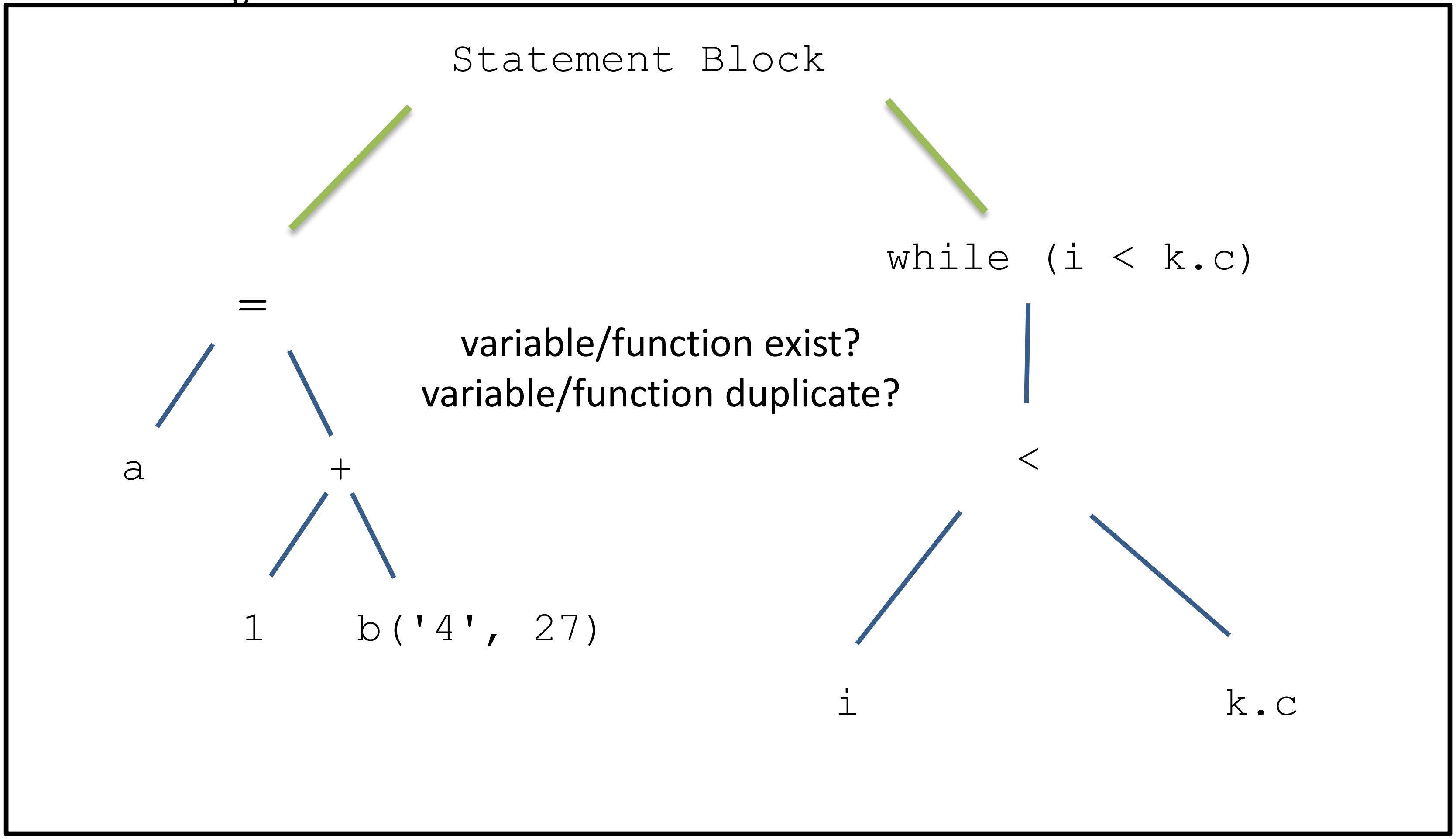
Creating the SAST

Function Index

Struct Index

Local Index

Function Ex()



Statement Block

=

a

+

1

b('4', 27)

while (i < k.c)

<

i

k.c

variable/function exist?
variable/function duplicate?

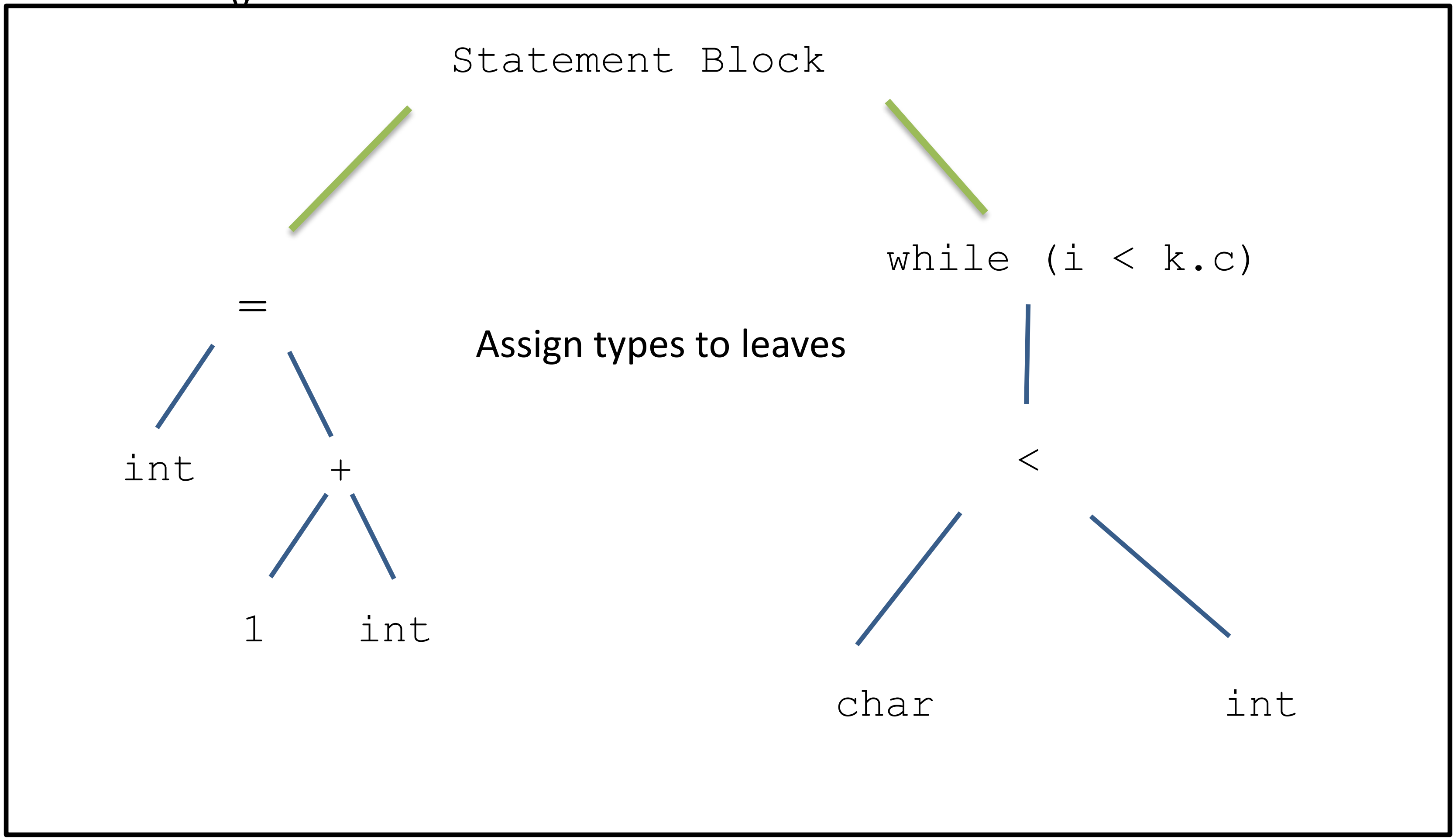
Creating the SAST

Function Index

Struct Index

Local Index

Function Ex()



Assign types to leaves

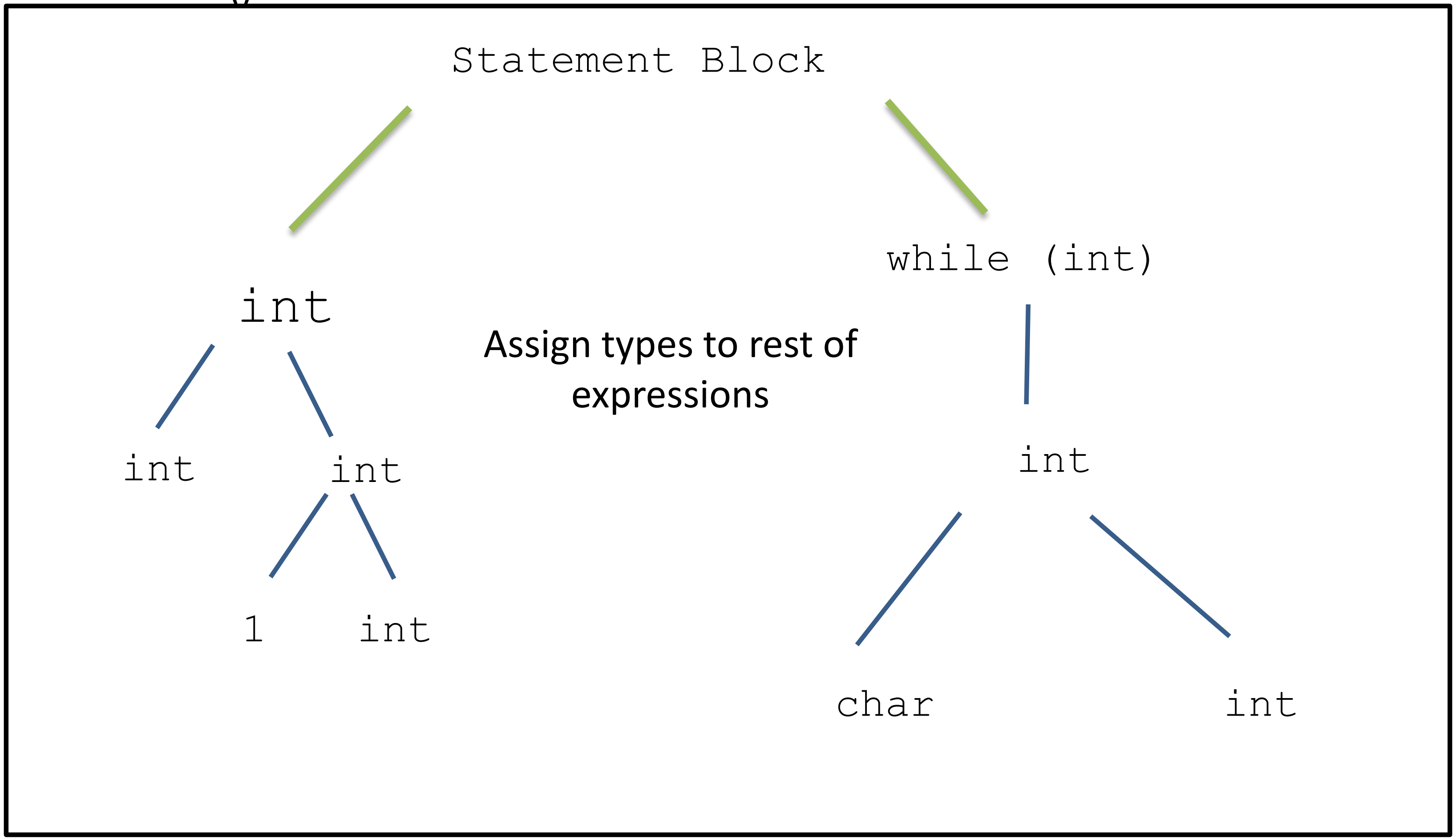
Creating the SAST

Function Index

Struct Index

Local Index

Function Ex()



Statement Block

int

int

int

1

int

while (int)

int

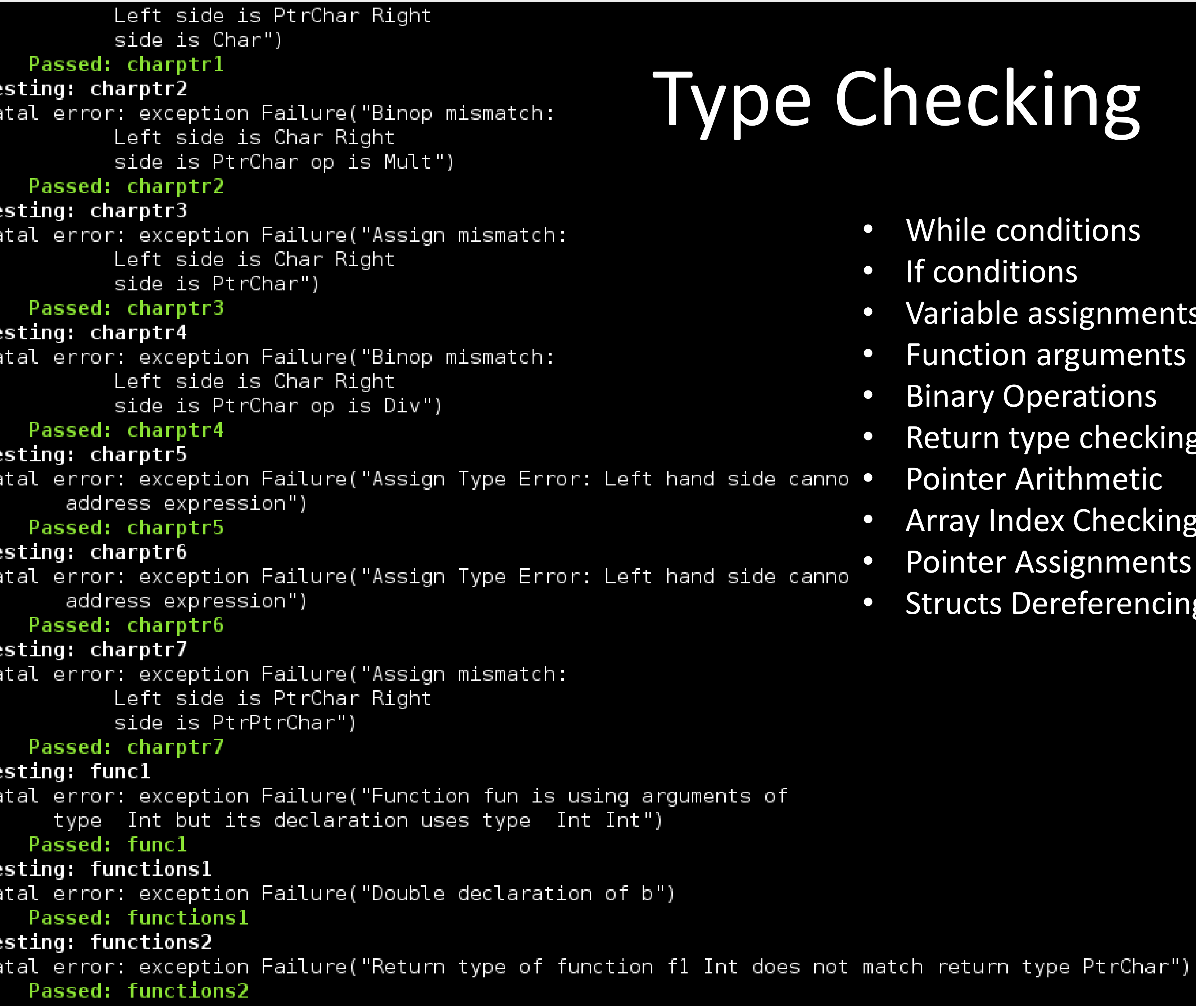
char

int

Assign types to rest of expressions

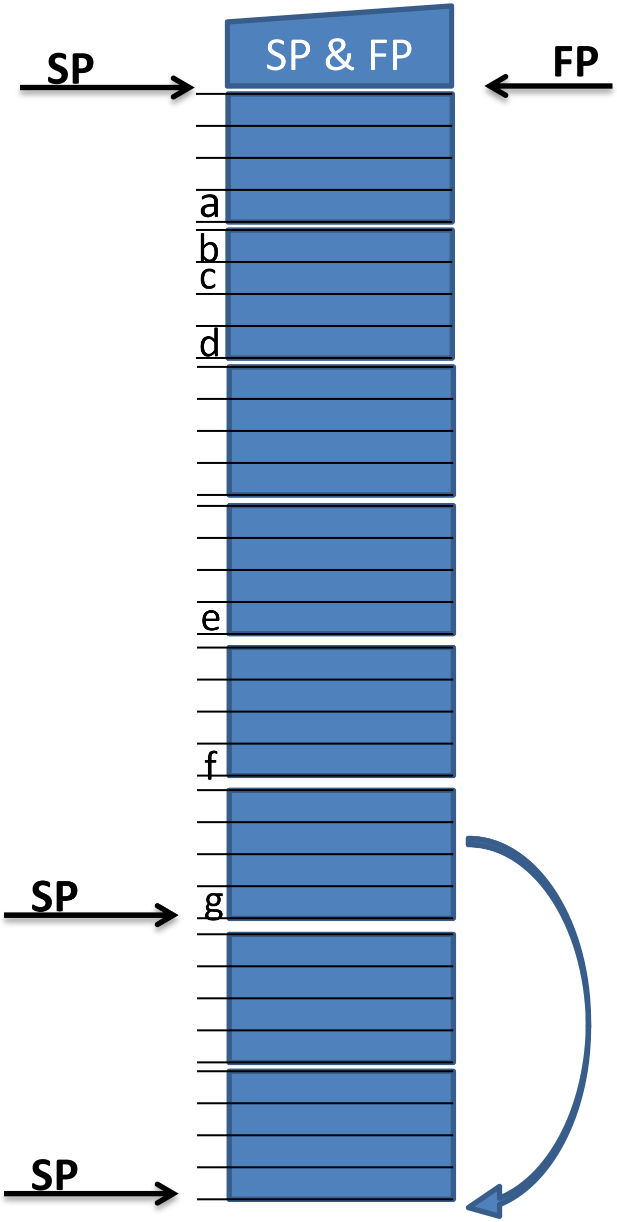
Type Checking

- While conditions
- If conditions
- Variable assignments
- Function arguments
- Binary Operations
- Return type checking
- Pointer Arithmetic
- Array Index Checking
- Pointer Assignments
- Structs Dereferencing



```
         Left side is PtrChar Right
         side is Char")
  Passed: charptr1
esting: charptr2
atal error: exception Failure("Binop mismatch:
         Left side is Char Right
         side is PtrChar op is Mult")
  Passed: charptr2
esting: charptr3
atal error: exception Failure("Assign mismatch:
         Left side is Char Right
         side is PtrChar")
  Passed: charptr3
esting: charptr4
atal error: exception Failure("Binop mismatch:
         Left side is Char Right
         side is PtrChar op is Div")
  Passed: charptr4
esting: charptr5
atal error: exception Failure("Assign Type Error: Left hand side canno
     address expression")
  Passed: charptr5
esting: charptr6
atal error: exception Failure("Assign Type Error: Left hand side canno
     address expression")
  Passed: charptr6
esting: charptr7
atal error: exception Failure("Assign mismatch:
         Left side is PtrChar Right
         side is PtrPtrChar")
  Passed: charptr7
esting: func1
atal error: exception Failure("Function fun is using arguments of
    type  Int but its declaration uses type  Int Int")
  Passed: func1
esting: functions1
atal error: exception Failure("Double declaration of b")
  Passed: functions1
esting: functions2
atal error: exception Failure("Return type of function f1 Int does not match return type PtrChar")
  Passed: functions2
```

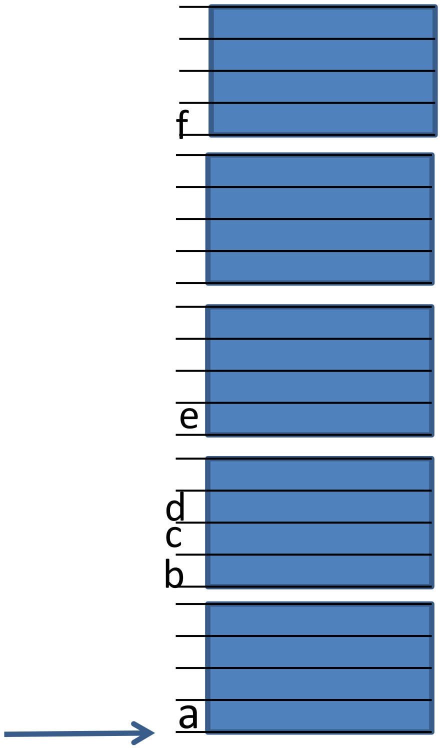
Variable Symbol Table



```
int a;
char b;
char c;
char d[2];
int e[2];
char *f;
int g[a]; (a=2)
```

varname	type	offset
a	[Int]	4
b	[Char]	5
c	[Char]	6
d	[Arr(2);Char]	8
e	[Arr(2);Int]	16
f	[Ptr;Char]	20
g	[Ptr;Int]	24

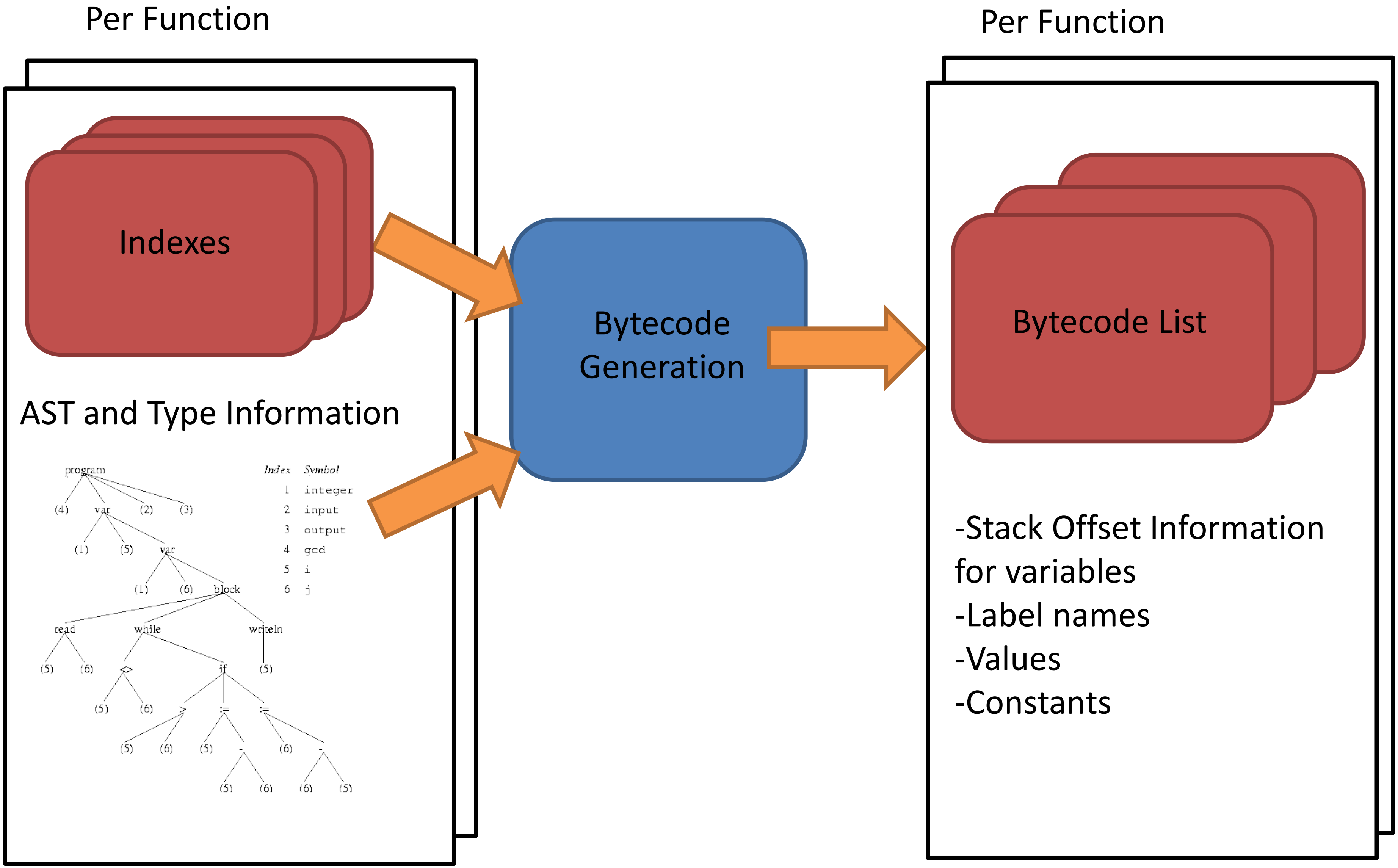
Structure Symbol Table



```
int a;
char b;
char c;
char d[2];
int e[2];
char *f;
```

varname	type	offset
a	[Int]	0
b	[Char]	5
c	[Char]	
d	[Arr(2);Char]	8
e	[Arr(2);Int]	16
f	[Ptr;Char]	20
~~g~~	~~[Ptr;Int]~~	~~24~~

Bytecode Generation



Per Function

Indexes

AST and Type Information

Index	*Symbol*
1	`integer`
2	`input`
3	`output`
4	`gcd`
5	`i`
6	`j`

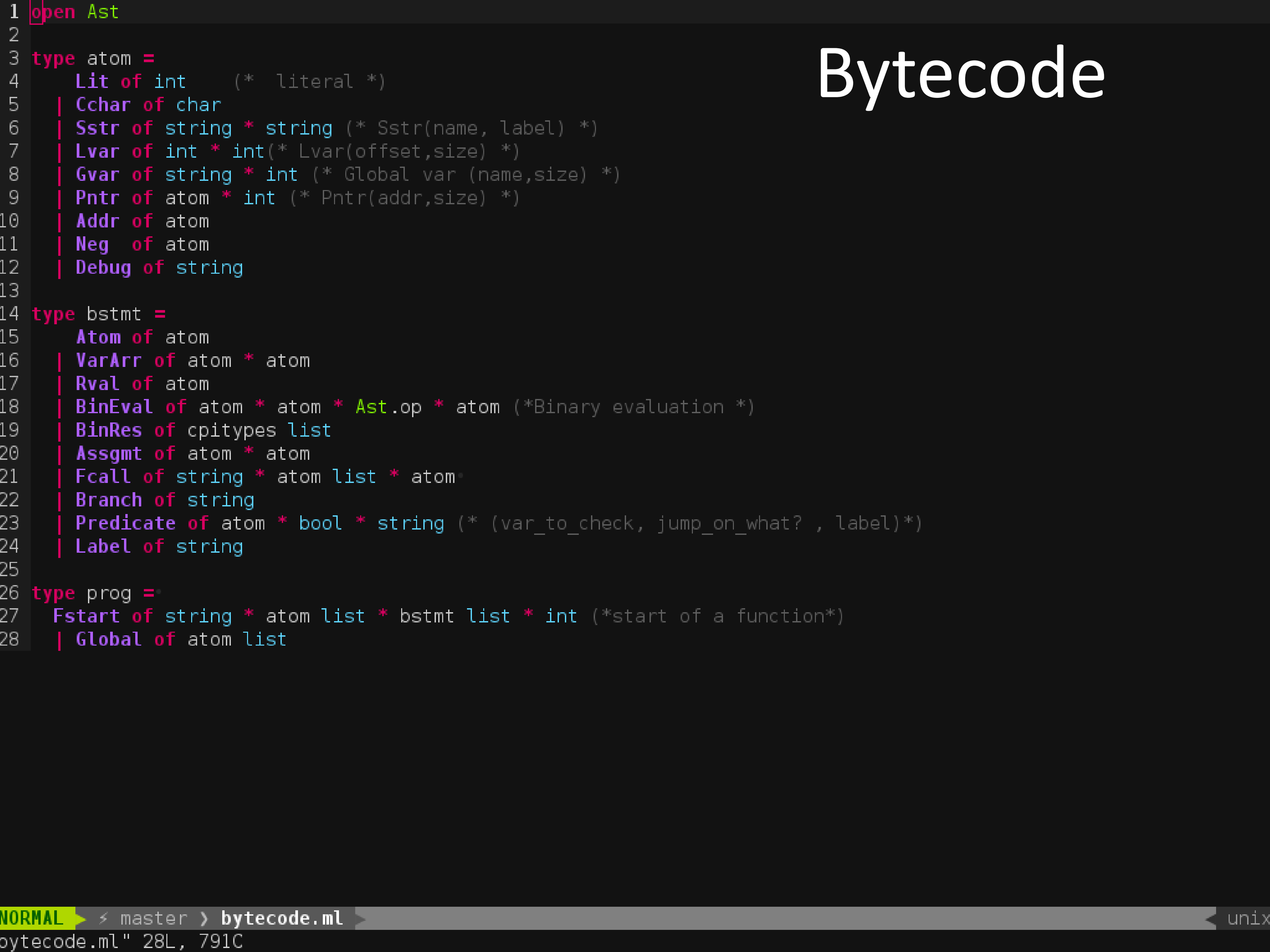
Bytecode Generation

Per Function

Bytecode List

-Stack Offset Information for variables
-Label names
-Values
-Constants

Bytecode



```
open Ast

type atom =
    Lit of int    (*  literal *)
  | Cchar of char
  | Sstr of string * string (* Sstr(name, label) *)
  | Lvar of int * int(* Lvar(offset,size) *)
  | Gvar of string * int (* Global var (name,size) *)
  | Pntr of atom * int (* Pntr(addr,size) *)
  | Addr of atom
  | Neg  of atom
  | Debug of string

type bstmt =
    Atom of atom
  | VarArr of atom * atom
  | Rval of atom
  | BinEval of atom * atom * Ast.op * atom (*Binary evaluation *)
  | BinRes of cpitypes list
  | Assgmt of atom * atom
  | Fcall of string * atom list * atom
  | Branch of string
  | Predicate of atom * bool * string (* (var_to_check, jump_on_what? , label)*)
  | Label of string

type prog =
  Fstart of string * atom list * bstmt list * int (*start of a function*)
  | Global of atom list
```

```
NORMAL  ⚡ master ❯ bytecode.ml                                         unix
oytecode.ml" 28L, 791C
```

The challenges

- Array offset calculation
  - arr[a+b+2]
- Pointer arithmetic
  - *(p+2)
  - *(2+a+p)
- Structure member offsets
  - s.a
  - s.a.c[3]
  - s->b
- All reduce to (base + offset) bytecode

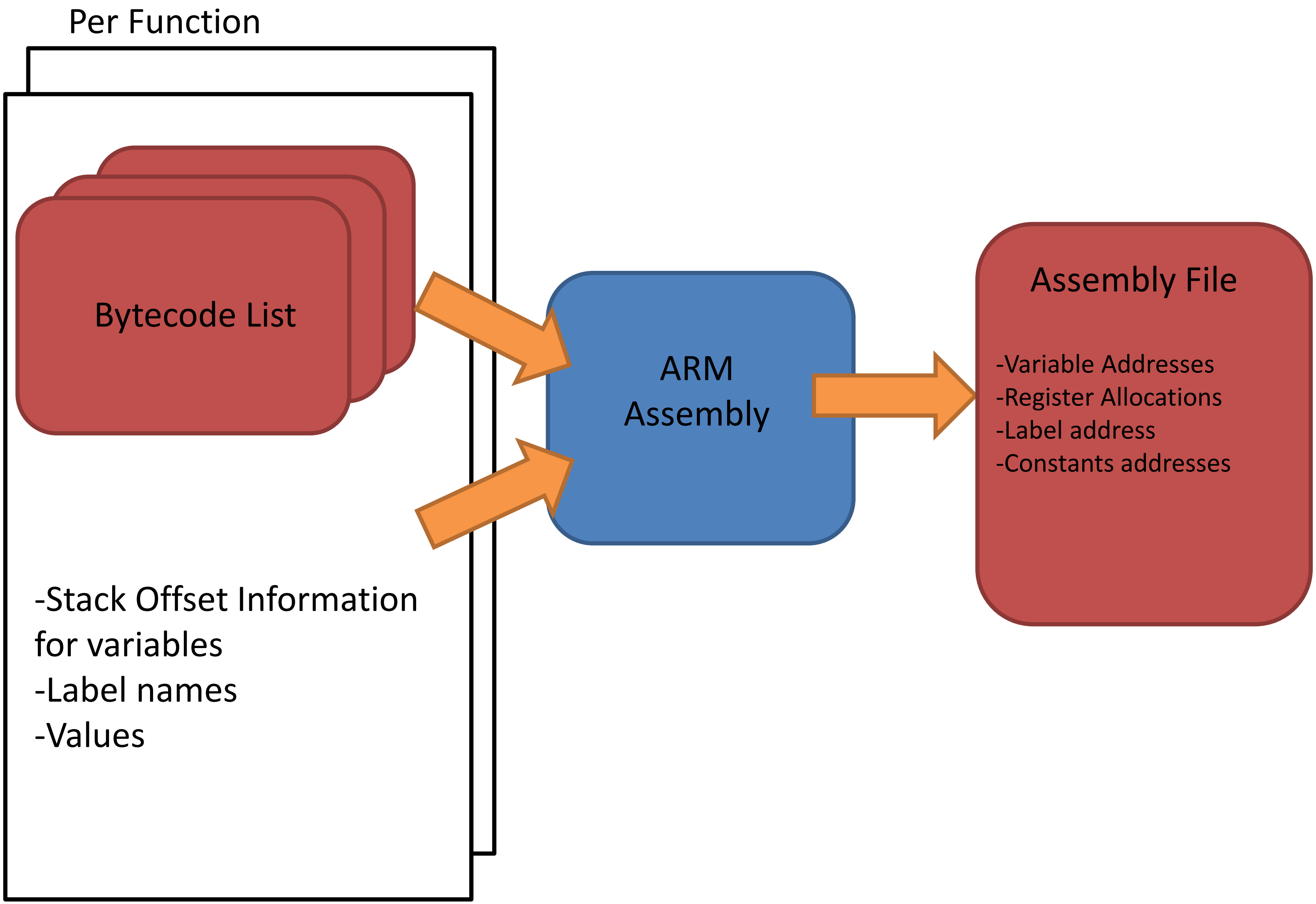
arr[a+b+2]

- BinEval(t1,a,+,b)
- BinEval(t2,t1,+,2)
- BinEval(t3,t2,*,4)
- BinEval(t4,Addr(arr),+,t3)
- Pntr(t4)

*(2+a+p)

- BinRes(Int);
  - BinEval(t1,2,+,a)
- BinRes(Ptr;Int)
  - BinEval(t2,t1,*,4)
  - BinEval(t3,p,+,t2)
- BinRes(Int)
  - Pntr(t3)

Arm Assembly Generation



Per Function

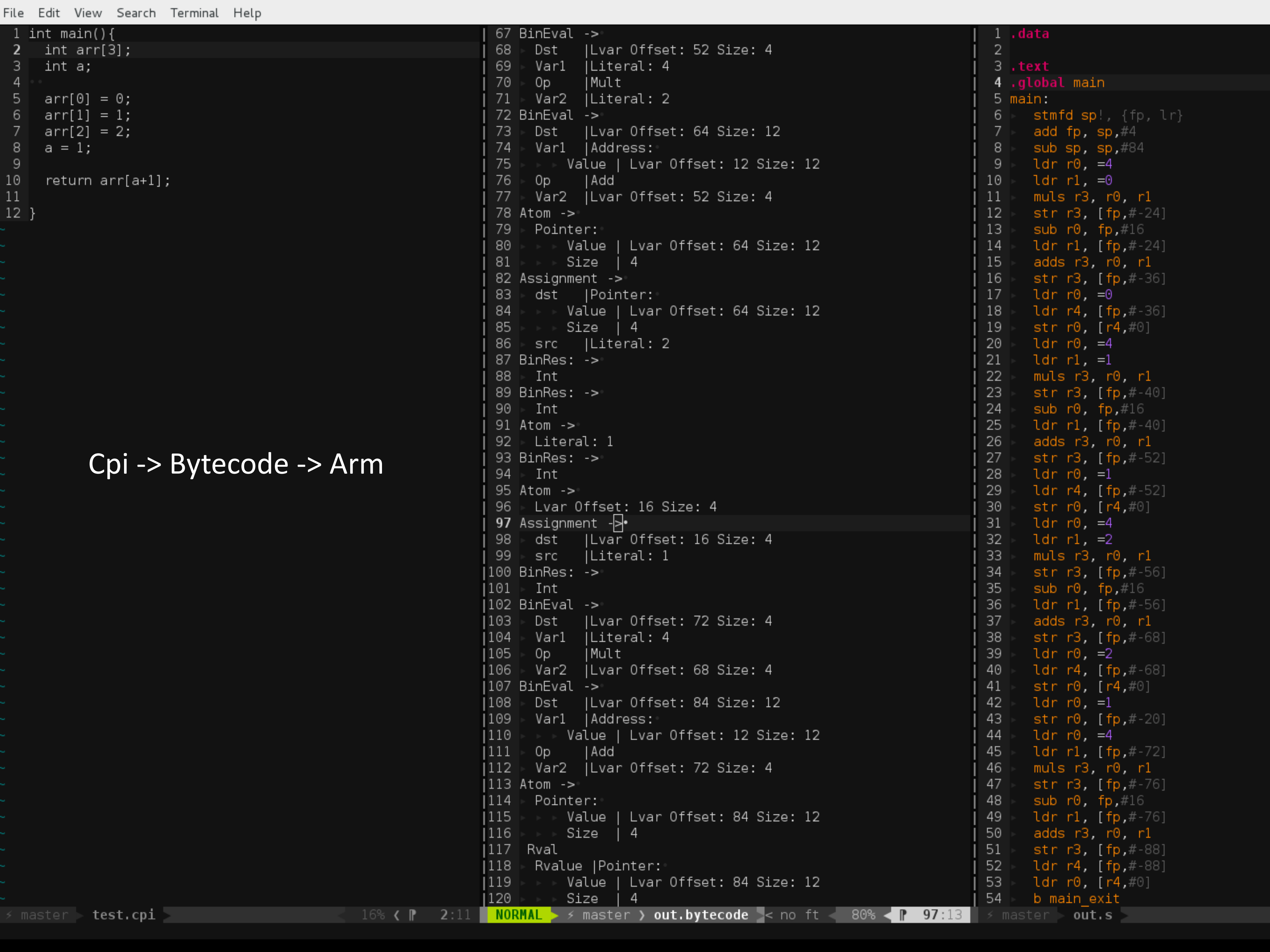
Bytecode List

-Stack Offset Information for variables
-Label names
-Values

ARM Assembly

Assembly File

-Variable Addresses
-Register Allocations
-Label address
-Constants addresses



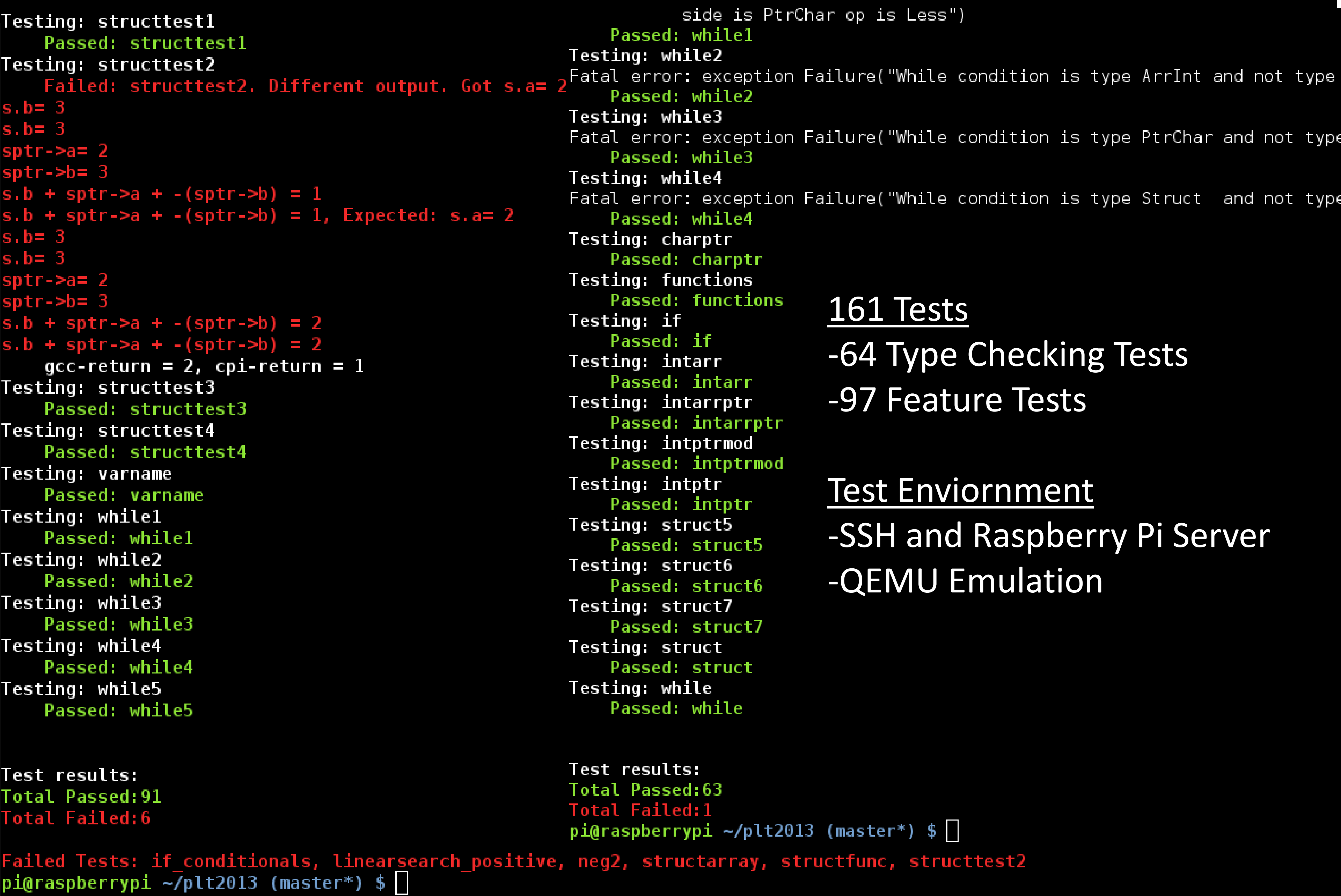
```
int main(){
  int arr[3];
  int a;

  arr[0] = 0;
  arr[1] = 1;
  arr[2] = 2;
  a = 1;

  return arr[a+1];

}
```

Cpi -> Bytecode -> Arm

```
BinEval ->
  Dst     |Lvar Offset: 52 Size: 4
  Var1    |Literal: 4
  Op      |Mult
  Var2    |Literal: 2
BinEval ->
  Dst     |Lvar Offset: 64 Size: 12
  Var1    |Address:
        Value | Lvar Offset: 12 Size: 12
  Op      |Add
  Var2    |Lvar Offset: 52 Size: 4
Atom ->
  Pointer:
        Value | Lvar Offset: 64 Size: 12
        Size  | 4
Assignment ->
  dst     |Pointer:
        Value | Lvar Offset: 64 Size: 12
        Size  | 4
  src     |Literal: 2
BinRes: ->
  Int
BinRes: ->
  Int
Atom ->
  Literal: 1
BinRes: ->
  Int
Atom ->
  Lvar Offset: 16 Size: 4
Assignment ->
  dst     |Lvar Offset: 16 Size: 4
  src     |Literal: 1
BinRes: ->
  Int
BinEval ->
  Dst     |Lvar Offset: 72 Size: 4
  Var1    |Literal: 4
  Op      |Mult
  Var2    |Lvar Offset: 68 Size: 4
BinEval ->
  Dst     |Lvar Offset: 84 Size: 12
  Var1    |Address:
        Value | Lvar Offset: 12 Size: 12
  Op      |Add
  Var2    |Lvar Offset: 72 Size: 4
Atom ->
  Pointer:
        Value | Lvar Offset: 84 Size: 12
        Size  | 4
 Rval
  Rvalue |Pointer:
        Value | Lvar Offset: 84 Size: 12
        Size  | 4
```

```
.data

.text
.global main
main:
    stmfd sp!, {fp, lr}
    add fp, sp,#4
    sub sp, sp,#84
    ldr r0, =4
    ldr r1, =0
    muls r3, r0, r1
    str r3, [fp,#-24]
    sub r0, fp,#16
    ldr r1, [fp,#-24]
    adds r3, r0, r1
    str r3, [fp,#-36]
    ldr r0, =0
    ldr r4, [fp,#-36]
    str r0, [r4,#0]
    ldr r0, =4
    ldr r1, =1
    muls r3, r0, r1
    str r3, [fp,#-40]
    sub r0, fp,#16
    ldr r1, [fp,#-40]
    adds r3, r0, r1
    str r3, [fp,#-52]
    ldr r0, =1
    ldr r4, [fp,#-52]
    str r0, [r4,#0]
    ldr r0, =4
    ldr r1, =2
    muls r3, r0, r1
    str r3, [fp,#-56]
    sub r0, fp,#16
    ldr r1, [fp,#-56]
    adds r3, r0, r1
    str r3, [fp,#-68]
    ldr r0, =2
    ldr r4, [fp,#-68]
    str r0, [r4,#0]
    ldr r0, =1
    str r0, [fp,#-20]
    ldr r0, =4
    ldr r1, [fp,#-72]
    muls r3, r0, r1
    str r3, [fp,#-76]
    sub r0, fp,#16
    ldr r1, [fp,#-76]
    adds r3, r0, r1
    str r3, [fp,#-88]
    ldr r4, [fp,#-88]
    ldr r0, [r4,#0]
    b main_exit
```

Testing



```
Testing: structtest1
    Passed: structtest1
Testing: structtest2
    Failed: structtest2. Different output. Got s.a= 2
s.b= 3
s.b= 3
sptr->a= 2
sptr->b= 3
s.b + sptr->a + -(sptr->b) = 1
s.b + sptr->a + -(sptr->b) = 1, Expected: s.a= 2
s.b= 3
s.b= 3
sptr->a= 2
sptr->b= 3
s.b + sptr->a + -(sptr->b) = 2
s.b + sptr->a + -(sptr->b) = 2
    gcc-return = 2, cpi-return = 1
Testing: structtest3
    Passed: structtest3
Testing: structtest4
    Passed: structtest4
Testing: varname
    Passed: varname
Testing: while1
    Passed: while1
Testing: while2
    Passed: while2
Testing: while3
    Passed: while3
Testing: while4
    Passed: while4
Testing: while5
    Passed: while5


Test results:
Total Passed:91
Total Failed:6

Failed Tests: if_conditionals, linearsearch_positive, neg2, structarray, structfunc, structtest2
pi@raspberrypi ~/plt2013 (master*) $ 
```

```
       side is PtrChar op is Less")
    Passed: while1
Testing: while2
Fatal error: exception Failure("While condition is type ArrInt and not type
    Passed: while2
Testing: while3
Fatal error: exception Failure("While condition is type PtrChar and not type
    Passed: while3
Testing: while4
Fatal error: exception Failure("While condition is type Struct  and not type
    Passed: while4
Testing: charptr
    Passed: charptr
Testing: functions
    Passed: functions
Testing: if
    Passed: if
Testing: intarr
    Passed: intarr
Testing: intarrptr
    Passed: intarrptr
Testing: intptrmod
    Passed: intptrmod
Testing: intptr
    Passed: intptr
Testing: struct5
    Passed: struct5
Testing: struct6
    Passed: struct6
Testing: struct7
    Passed: struct7
Testing: struct
    Passed: struct
Testing: while
    Passed: while


Test results:
Total Passed:63
Total Failed:1
pi@raspberrypi ~/plt2013 (master*) $ 
```

161 Tests

-64 Type Checking Tests

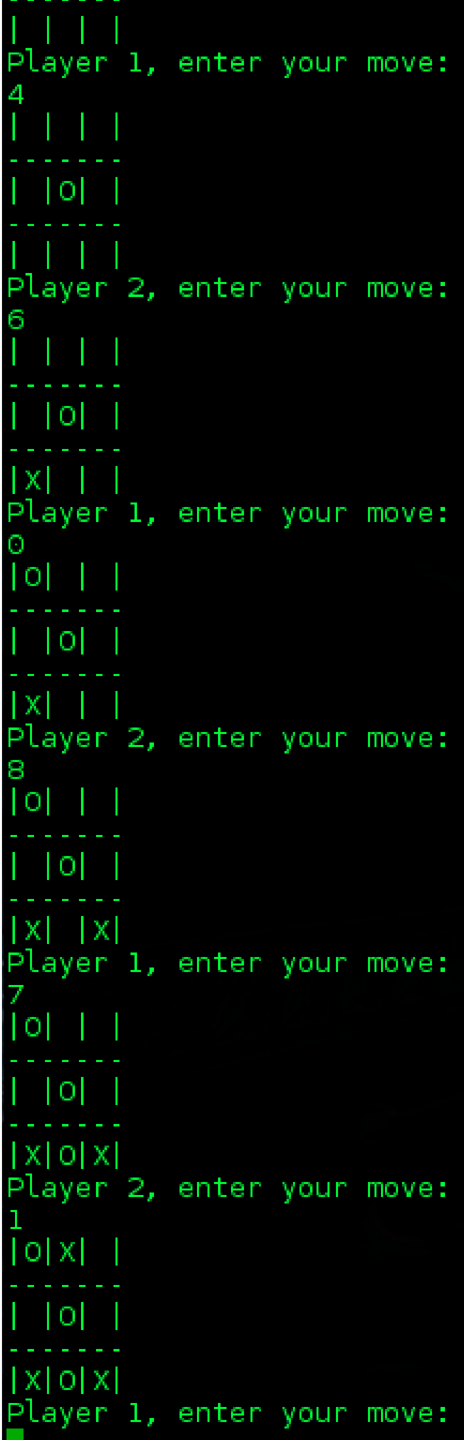
-97 Feature Tests

Test Enviornment

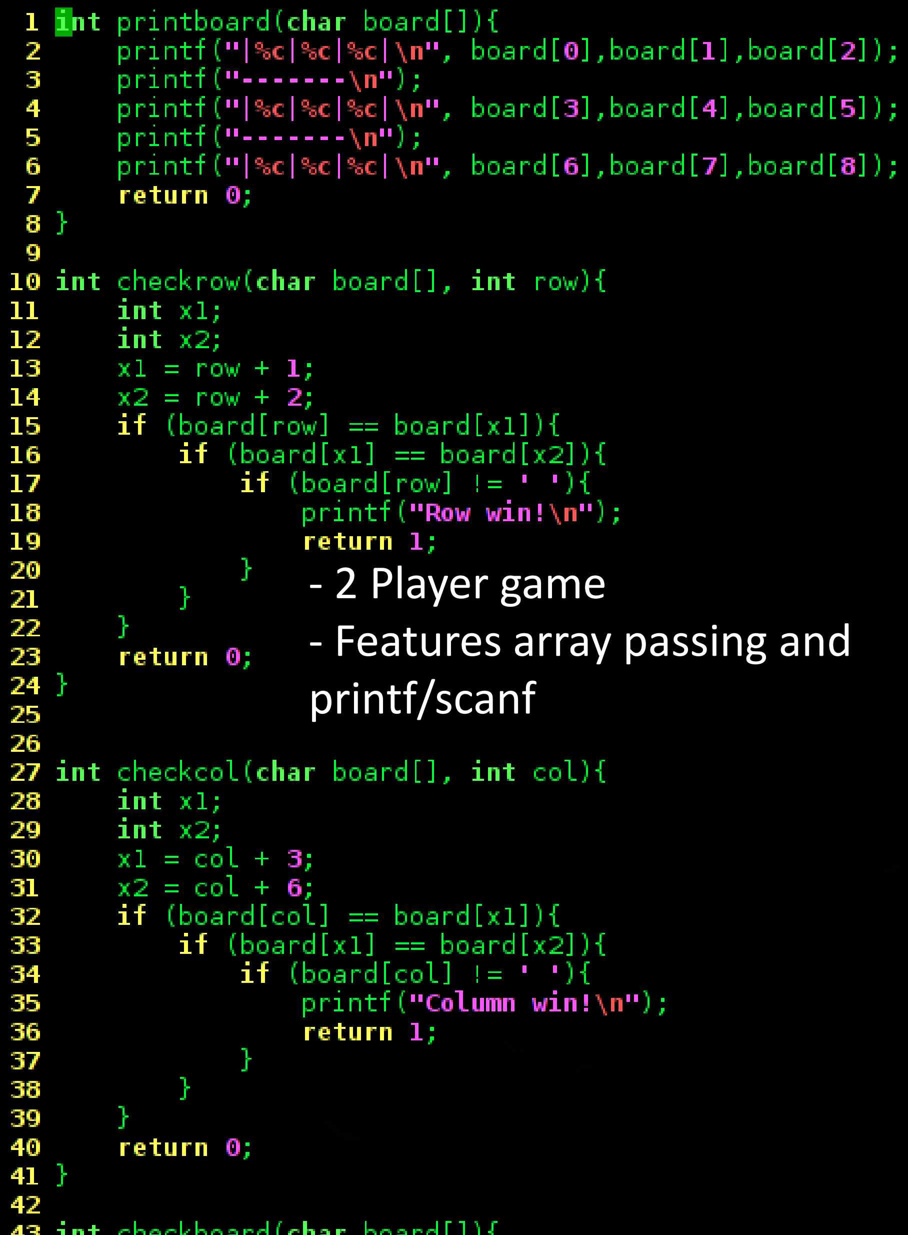
-SSH and Raspberry Pi Server

-QEMU Emulation

Example: Tic-Tac-Toe

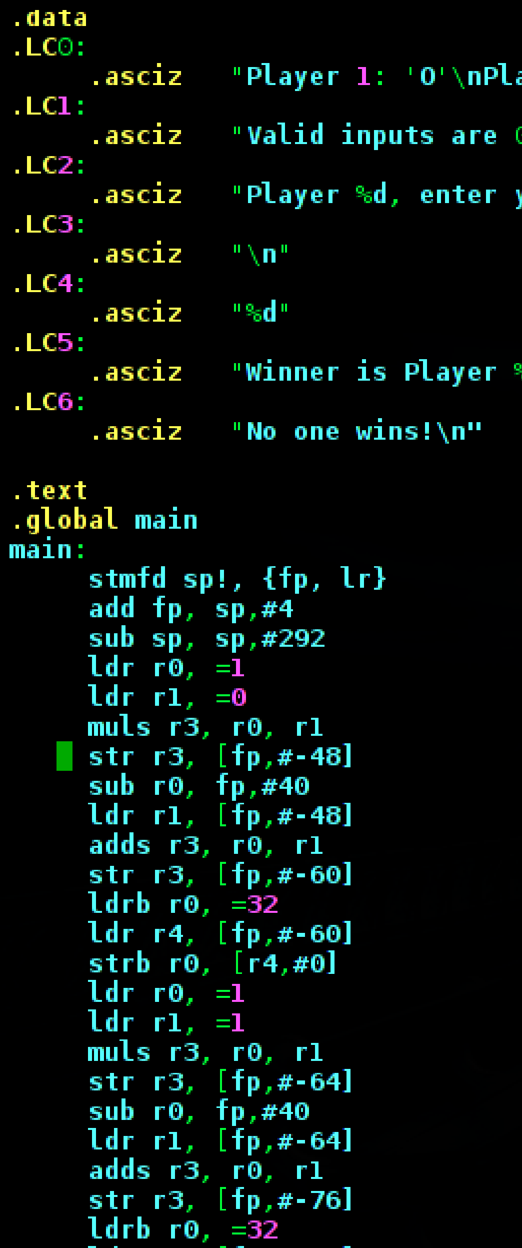


```
| | | |
Player 1, enter your move:
4
| | | |
-------
| |O| |
-------
| | | |
Player 2, enter your move:
6
| | | |
-------
| |O| |
-------
|X| | |
Player 1, enter your move:
0
|O| | |
-------
| |O| |
-------
|X| | |
Player 2, enter your move:
8
|O| | |
-------
| |O| |
-------
|X| |X|
Player 1, enter your move:
7
|O| | |
-------
| |O| |
-------
|X|O|X|
Player 2, enter your move:
1
|O|X| |
-------
| |O| |
-------
|X|O|X|
Player 1, enter your move:
```



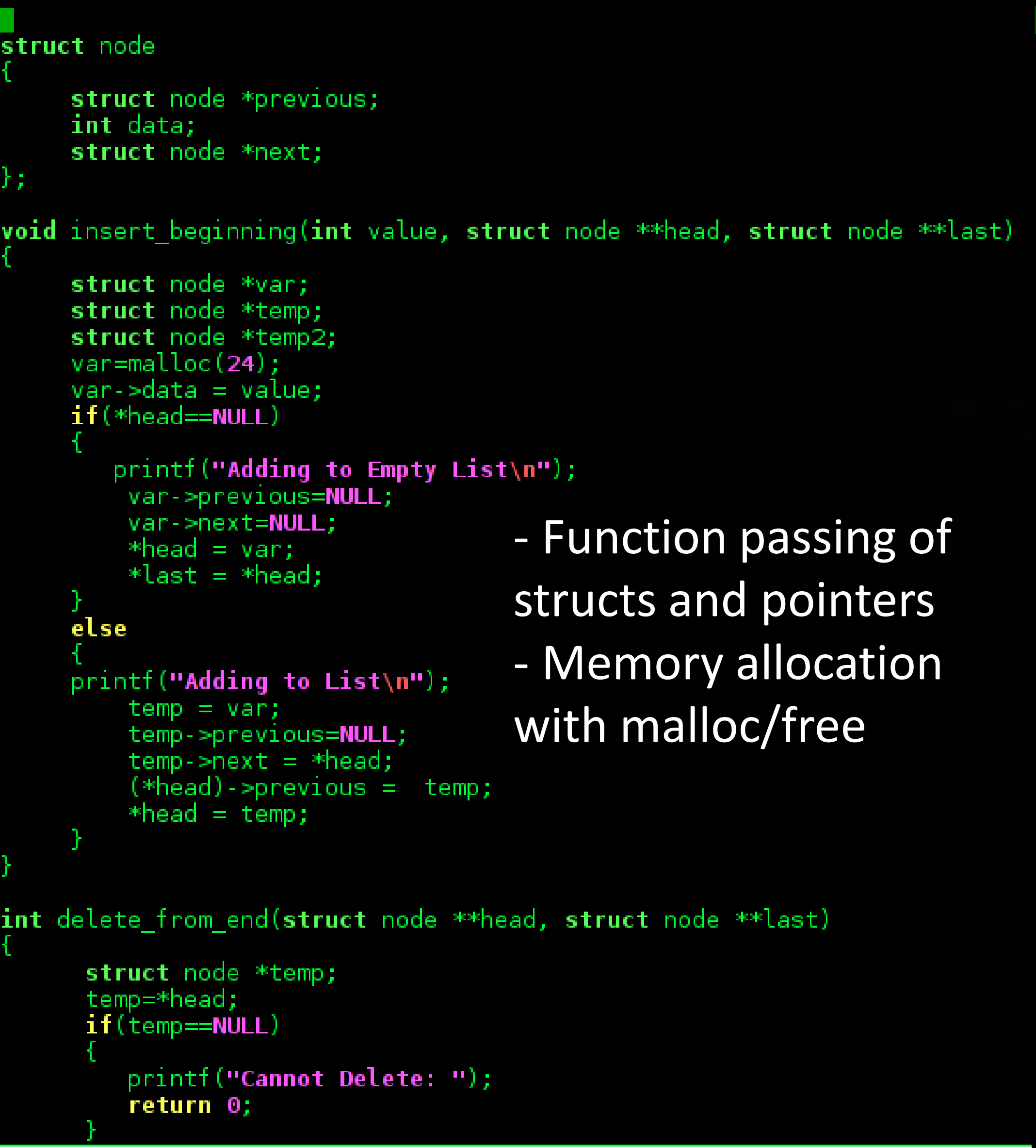
```c
int printboard(char board[]){
    printf("|%c|%c|%c|\n", board[0],board[1],board[2]);
    printf("-------\n");
    printf("|%c|%c|%c|\n", board[3],board[4],board[5]);
    printf("-------\n");
    printf("|%c|%c|%c|\n", board[6],board[7],board[8]);
    return 0;
}

int checkrow(char board[], int row){
    int x1;
    int x2;
    x1 = row + 1;
    x2 = row + 2;
    if (board[row] == board[x1]){
        if (board[x1] == board[x2]){
            if (board[row] != ' '){
                printf("Row win!\n");
                return 1;
            }
        }
    }
    return 0;
}


int checkcol(char board[], int col){
    int x1;
    int x2;
    x1 = col + 3;
    x2 = col + 6;
    if (board[col] == board[x1]){
        if (board[x1] == board[x2]){
            if (board[col] != ' '){
                printf("Column win!\n");
                return 1;
            }
        }
    }
    return 0;
}

int checkboard(char board[]){
```

- 2 Player game
- Features array passing and printf/scanf



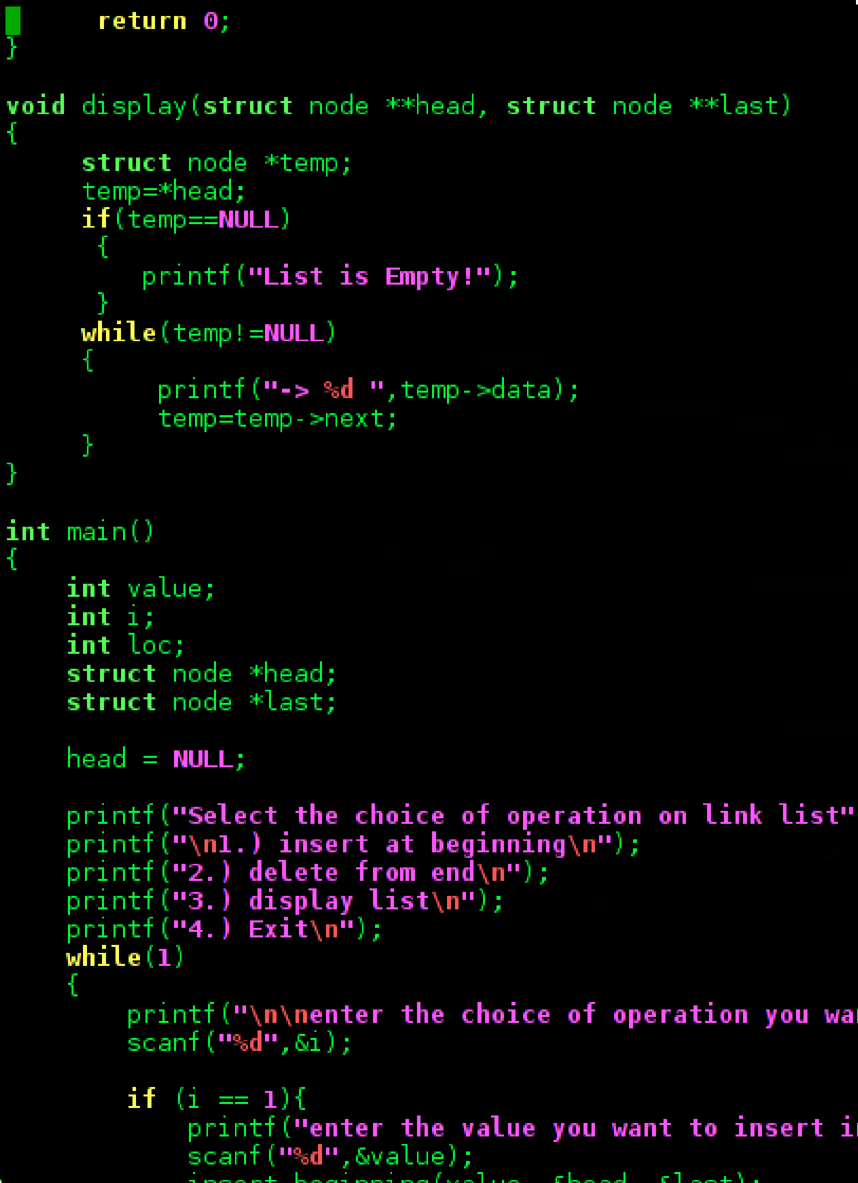
```
.data
.LC0:
        .asciz   "Player 1: 'O'\nPla
.LC1:
        .asciz   "Valid inputs are (
.LC2:
        .asciz   "Player %d, enter y
.LC3:
        .asciz   "\n"
.LC4:
        .asciz   "%d"
.LC5:
        .asciz   "Winner is Player %
.LC6:
        .asciz   "No one wins!\n"

.text
.global main
main:
        stmfd sp!, {fp, lr}
        add fp, sp,#4
        sub sp, sp,#292
        ldr r0, =1
        ldr r1, =0
        muls r3, r0, r1
        str r3, [fp,#-48]
        sub r0, fp,#40
        ldr r1, [fp,#-48]
        adds r3, r0, r1
        str r3, [fp,#-60]
        ldrb r0, =32
        ldr r4, [fp,#-60]
        strb r0, [r4,#0]
        ldr r0, =1
        ldr r1, =1
        muls r3, r0, r1
        str r3, [fp,#-64]
        sub r0, fp,#40
        ldr r1, [fp,#-64]
        adds r3, r0, r1
        str r3, [fp,#-76]
        ldrb r0, =32
```

Example: Linked List



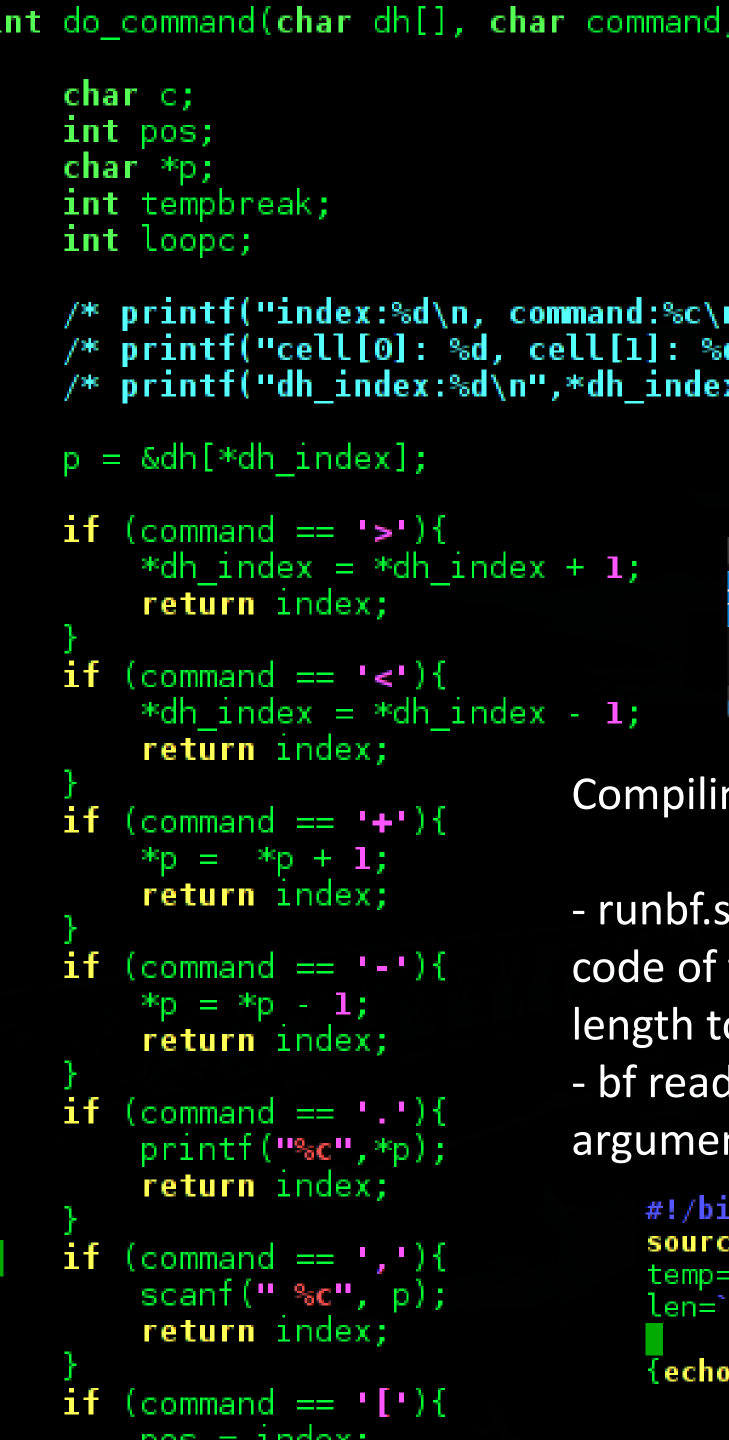
```
struct node
{
     struct node *previous;
     int data;
     struct node *next;
};

void insert_beginning(int value, struct node **head, struct node **last)
{
     struct node *var;
     struct node *temp;
     struct node *temp2;
     var=malloc(24);
     var->data = value;
     if(*head==NULL)
     {
        printf("Adding to Empty List\n");
         var->previous=NULL;
         var->next=NULL;
         *head = var;
         *last = *head;
     }
     else
     {
     printf("Adding to List\n");
         temp = var;
         temp->previous=NULL;
         temp->next = *head;
         (*head)->previous =  temp;
         *head = temp;
     }
}

int delete_from_end(struct node **head, struct node **last)
{
      struct node *temp;
      temp=*head;
      if(temp==NULL)
      {
         printf("Cannot Delete: ");
         return 0;
      }
```

- Function passing of structs and pointers
- Memory allocation with malloc/free

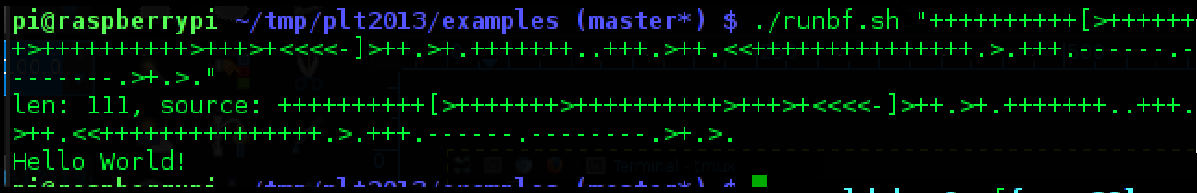


```
     return 0;
}

void display(struct node **head, struct node **last)
{
     struct node *temp;
     temp=*head;
     if(temp==NULL)
      {
         printf("List is Empty!");
      }
     while(temp!=NULL)
     {
          printf("-> %d ",temp->data);
          temp=temp->next;
     }
}

int main()
{
    int value;
    int i;
    int loc;
    struct node *head;
    struct node *last;

    head = NULL;

    printf("Select the choice of operation on link list"
    printf("\n1.) insert at beginning\n");
    printf("2.) delete from end\n");
    printf("3.) display list\n");
    printf("4.) Exit\n");
    while(1)
    {
        printf("\n\nenter the choice of operation you wa
        scanf("%d",&i);

        if (i == 1){
            printf("enter the value you want to insert i
            scanf("%d",&value);
            insert_beginning(value, &head, &last);
```

Example: Brainfuck Interpreter



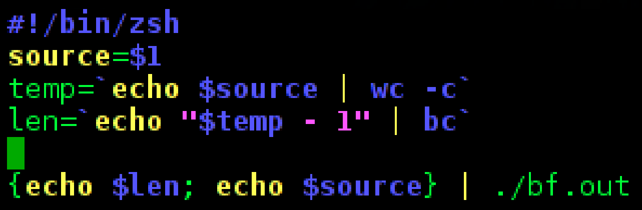
```
nt do_command(char dh[], char command, char source[], in

   char c;
   int pos;
   char *p;
   int tempbreak;
   int loopc;

   /* printf("index:%d\n, command:%c\n",index,command);
   /* printf("cell[0]: %d, cell[1]: %d\n",dh[0], dh[1]);
   /* printf("dh_index:%d\n",*dh_index); */

   p = &dh[*dh_index];

   if (command == '>'){
       *dh_index = *dh_index + 1;
       return index;
   }
   if (command == '<'){
       *dh_index = *dh_index - 1;
       return index;
   }
   if (command == '+'){
       *p =  *p + 1;
       return index;
   }
   if (command == '-'){
       *p = *p - 1;
       return index;
   }
   if (command == '.'){
       printf("%c",*p);
       return index;
   }
   if (command == ','){
       scanf(" %c", p);
       return index;
   }
   if (command == '['){
       pos = index;
```



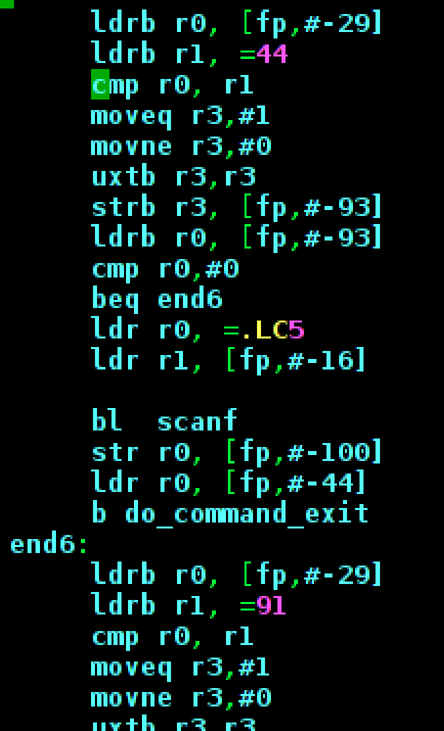
```
pi@raspberrypi ~/tmp/plt2013/examples (master*) $ ./runbf.sh "++++++++++[>++++++
+>++++++++++>+++>+<<<<-]>++.>+.+++++++..+++.>++.<<+++++++++++++++.>.+++.------.-
-------.>+.>."
len: 111, source: ++++++++++[>+++++++>++++++++++>+++>+<<<<-]>++.>+.+++++++..+++.
>++.<<+++++++++++++++.>.+++.------.--------.>+.>.
Hello World!
pi@raspberrypi ~/tmp/plt2013/examples (master*) $
```

Compiling an interpreter?? Yes!

- runbf.sh is used to pass the source code of the bf program along with its length to the bf interpreter
- bf reads the two command line arguments through scanf



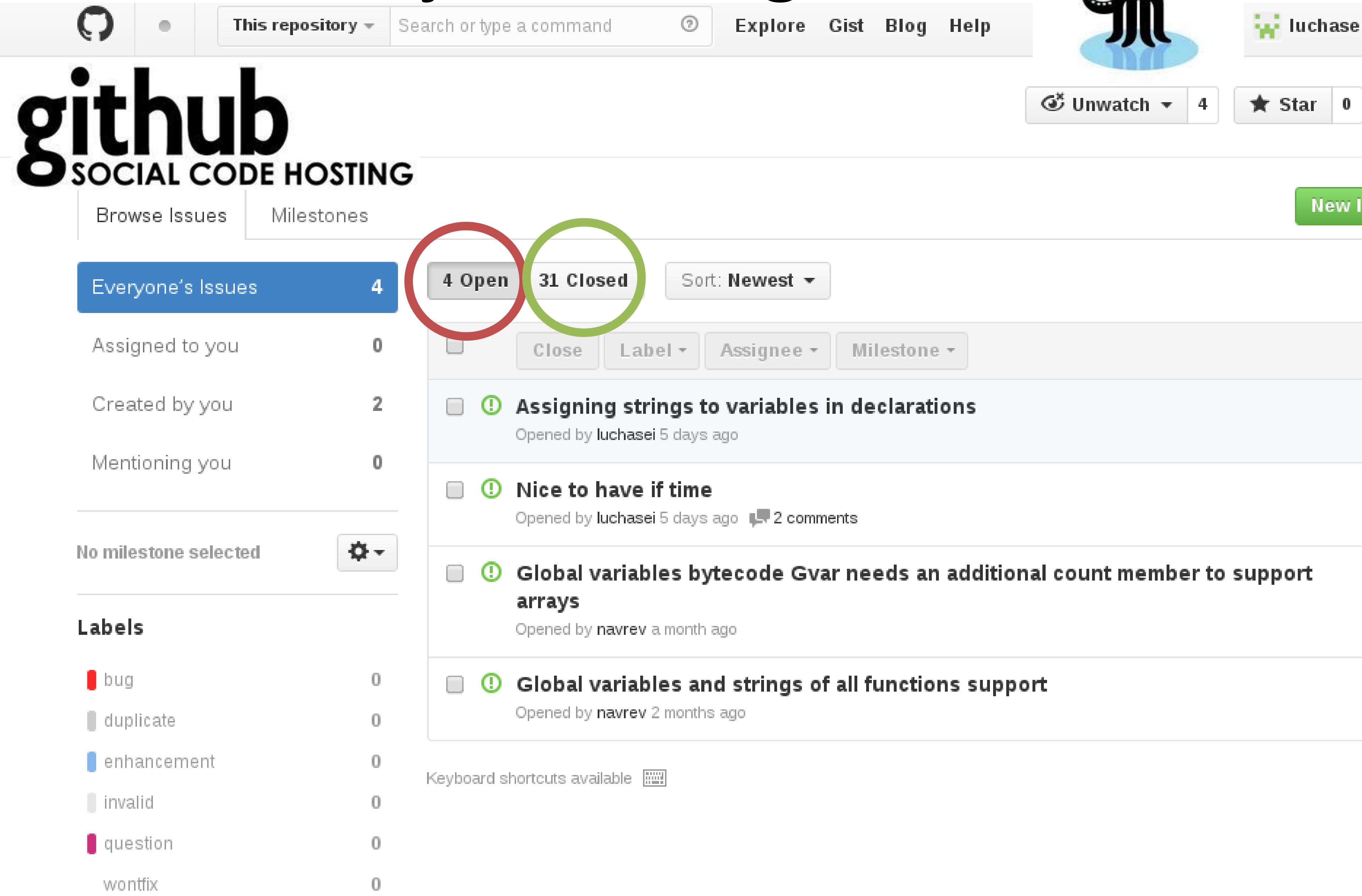
```
#!/bin/zsh
source=$1
temp=`echo $source | wc -c`
len=`echo "$temp - 1" | bc`

{echo $len; echo $source} | ./bf.out
```



```
          ldrb r0, [fp,#-29]
          ldrb r1, =44
          cmp r0, r1
          moveq r3,#1
          movne r3,#0
          uxtb r3,r3
          strb r3, [fp,#-93]
          ldrb r0, [fp,#-93]
          cmp r0,#0
          beq end6
          ldr r0, =.LC5
          ldr r1, [fp,#-16]

          bl  scanf
          str r0, [fp,#-100]
          ldr r0, [fp,#-44]
          b do_command_exit
end6:
          ldrb r0, [fp,#-29]
          ldrb r1, =91
          cmp r0, r1
          moveq r3,#1
          movne r3,#0
          uxtb r3,r3
```

Project Management

github
SOCIAL CODE HOSTING



4 Open | 31 Closed | Sort: Newest

Everyone's Issues	4
Assigned to you	0
Created by you	2
Mentioning you	0

No milestone selected

□ Assigning strings to variables in declarations
Opened by luchasei 5 days ago

□ Nice to have if time
Opened by luchasei 5 days ago 2 comments

□ Global variables bytecode Gvar needs an additional count member to support arrays
Opened by navrev a month ago

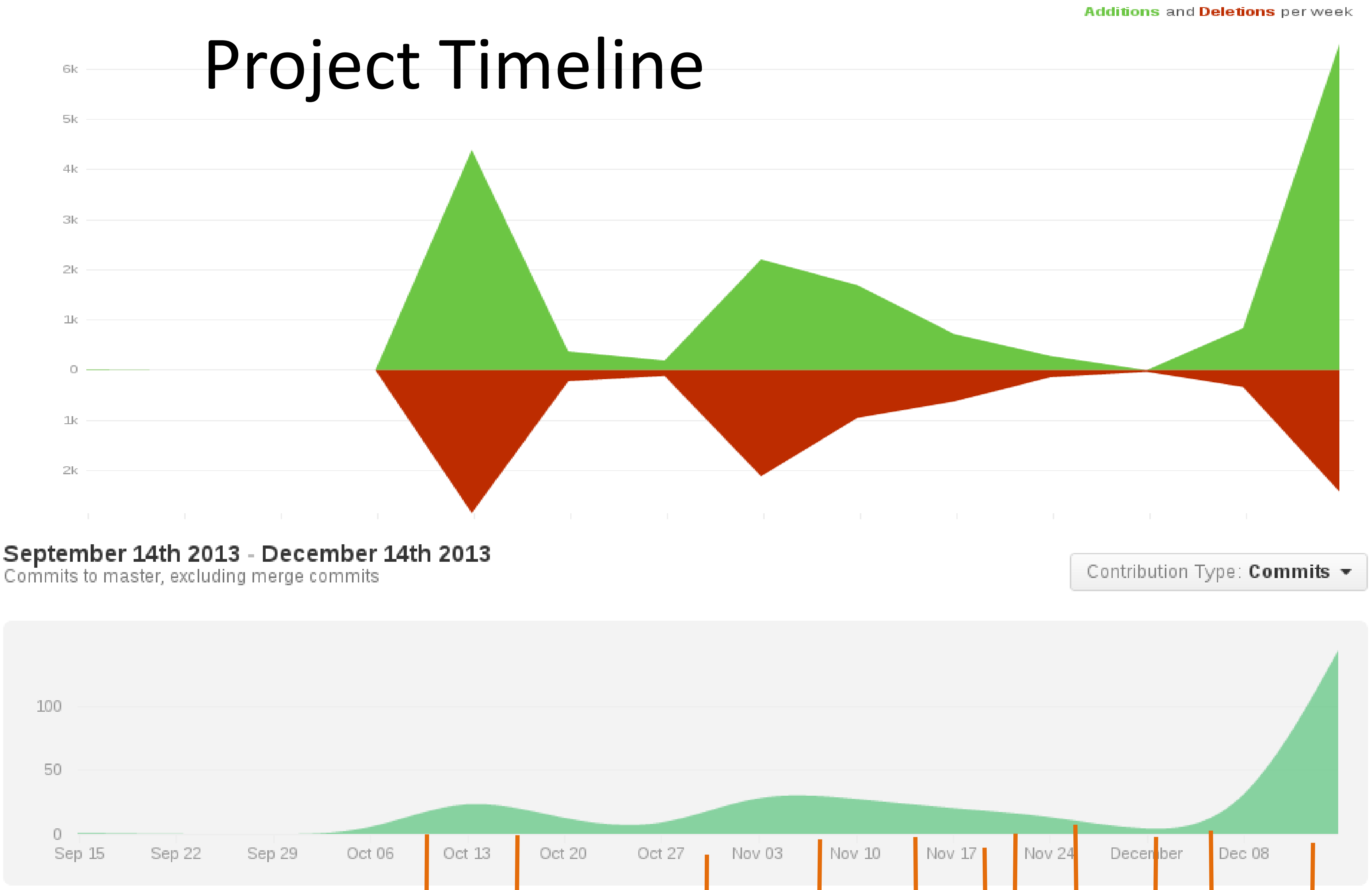
□ Global variables and strings of all functions support
Opened by navrev 2 months ago

Keyboard shortcuts available

Labels

🟥 bug	0
duplicate	0
enhancement	0
invalid	0
question	0
wontfix	0

Project Timeline



September 14th 2013 - December 14th 2013
Commits to master, excluding merge commits

Contribution Type: **Commits**

Scanner
Parser
ARM Binops

Hello World

Bytecode
Generation

Test
Framework

Offset Calc Rework

return

Pointers

while
if

structs

feature
tests

SAST
Tests
Bug Fix

Contributions

- Naveen Revanna - Architecture Czar, Bytecode Generation
- Eddy Garcia - Type Checking, Test Case Generation, External functions
- Sean Yeh - Test suite, Example programs, bug fixes
- Niket Kandya - Scanner/Parser, Scalar Types and Functions, Design

Lessons Learned

- Naveen Revanna - Spend sufficient time in deciding a scalable architecture at early stages. Don't trust your developer self. Document code sufficiently. A good test infrastructure can save you loads of time.
- Eddy Garcia - Pattern matching should be a feature available in all languages. Regression tests are wonderful.
- Sean Yeh - Next time I will not write test suite script in BASH. Nevertheless, the testing framework turned out pretty well.
- Niket Kandya - Time spent on good design is time saved. Functional Programming is a clean approach. Compilers are fun.