

Compiladores¹ Geração de Código II

Capítulo 11 - "Compiladores: Da Teoria à Prática"

Prof. Alberto Abad

IST - Universidade de Lisboa

2021/2022

¹Slides adaptados de Prof. Pedro T. Monteiro (2017/2018)

Consultem o Manual de Referência!

https://web.tecnico.ulisboa.pt/~david.matos/w/pt/index.php/Postfix_Reference_Guide

Frame Pointer



Frame Pointer (FP) - indica na pilha o contexto local de uma função

Frame Pointer



Frame Pointer (FP) - indica na pilha o contexto local de uma função

- Na entrada de função:
 - o FP guarda o valor do SP

Frame Pointer



Frame Pointer (FP) - indica na pilha o contexto local de uma função

- Na entrada de função:
 - o FP guarda o valor do SP
- Durante a execução:
 - o SP pode variar
 - o FP mantém-se constante
 - os argumentos da função e as variáveis locais podem ser acedidas através de um offset fixo



Variáveis locais

- Existem apenas no contexto de uma função
- Residem na pilha
- Acessíveis através de um offset
- offset é negativo relativamente ao FP



Variáveis locais

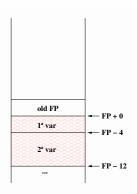
- Existem apenas no contexto de uma função
- Residem na pilha
- Acessíveis através de um offset
- offset é negativo relativamente ao FP





Variáveis locais

- Existem apenas no contexto de uma função
- Residem na pilha
- Acessíveis através de um offset
- offset é negativo relativamente ao FP

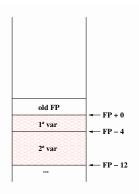


LOCAL - carrega o endereço de um offset para a pilha



Variáveis locais

- Existem apenas no contexto de uma função
- Residem na pilha
- Acessíveis através de um offset
- offset é negativo relativamente ao FP



LOCAL - carrega o endereço de um *offset* para a pilha Depois o endereço é lido/escrito da forma habitual:

- LDINT, LDDOUBLE, ...
- STINT, STDOUBLE, ...



Endereçamento indirecto (pilha)

Endereçamento de variáveis locais



Endereçamento indirecto (pilha)

Endereçamento de variáveis locais

```
v = 2;
INT 2
DUP32
LOCAL -4
STINT
TRASH 4
```



Endereçamento indirecto (pilha)

Endereçamento de variáveis locais

v = 2;	v = x;
INT 2	LOCAL -8
	LDINT
DUP32	DUP32
LOCAL -4	DUP32
	LOCAL -4
STINT	STINT
TRASH 4	
	TDACH /



```
Definição de função
void fname(int a, char * b) {
  int i;
  double d;
  ...
}
```



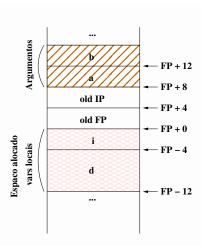
```
Definição de função
void fname(int a, char * b) {
  int i;
  double d;
   . . .
TEXT
ALIGN
GLOBAL fname, FUNC
LABEL fname
ENTER?
```



```
Definição de função
void fname(int a, char * b) {
  int i;
  double d;
   . . .
TEXT
AT.TGN
GLOBAL fname, FUNC
LABEL fname
ENTER 12
;; evaluate function body
LEAVE
RET
```

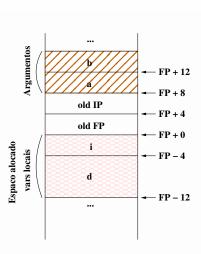


```
Definição de função
void fname(int a, char * b) {
  int i;
  double d;
   . . .
TEXT
AT.TGN
GLOBAL fname, FUNC
LABEL fname
ENTER 12
;; evaluate function body
LEAVE
RET
```





```
Definição de função
void fname(int a, char * b) {
  int i;
  double d;
   . . .
TEXT
AT.TGN
GLOBAL fname, FUNC
LABEL fname
ENTER 12
;; evaluate function body
LEAVE
RET
```



Q: Como o copilador sabe o tamanho do ENTER?



- LEAVE restaura o FP anterior e apaga a pilha até ao FP
- RET restaura o IP anterior e apaga-o da pilha



- LEAVE restaura o FP anterior e apaga a pilha até ao FP
- RET restaura o IP anterior e apaga-o da pilha

Q: Como é que o valor de retorno é passado?



- LEAVE restaura o FP anterior e apaga a pilha até ao FP
- RET restaura o IP anterior e apaga-o da pilha

Q: Como é que o valor de retorno é passado?

- STFVAL retira o valor da pilha para o registo EAX/ST0
- LDFVAL retira o valor do registo EAX/STO para a pilha



Declaração de variável funcional (global)

```
var vname = (int a, [text] b) -> void :
  int i
  double d
  ...
```



```
Declaração de variável funcional (global)
var vname = (int a, [text] b) -> void :
  int i
  double d
   . . .
                             TEXT F123
DATA
                             AT.TGN
AT.TGN
                             LABEL F123
GLOBAL vname, OBJ
                             ENTER 12
LABEL vname
                             ;; evaluate function body
SADDR F123
                             LEAVE
. . .
                             RFT
```



```
Declaração de variável funcional (global) var vname = (int a, [text] b) -> void :
```

int i double d

• • •

DATA TEXT _F123

ALIGN ALIGN

GLOBAL vname, OBJ LABEL _F123

LABEL vname ENTER 12

SADDR _F123 ;; evaluate function body

LEAVE RFT

Q: Como seria com variáveis funcionais locais?



Invocação de variável funcional (global)

```
int<int, int> vname
...
vname(0, 1)
...
```



```
Invocação de variável funcional (global)
int<int, int> vname
vname(0, 1)
INT 1
INT 0
ADDR vname
LDINT
BRANCH
```

TRASH 4

. . .



```
Invocação de variável funcional (global)
int<int, int> vname
vname(0, 1)
INT 1
INT 0
ADDR vname
LDINT
BRANCH
TRASH 4
```

Q: Como seria com variáveis funcionais locais?

. . .



```
Invocação de variável funcional (global)
int<int, int> vname
vname(0, 1)
INT 1
INT 0
ADDR vname
LDINT
BRANCH
TRASH 4
. . .
```

- Q: Como seria com variáveis funcionais locais?
- Q: Como seria com funções externas foreign?



Um vector é um ponteiro para a base acrescido de um deslocamento



Um vector é um ponteiro para a base acrescido de um deslocamento

A soma de um ponteiro p com um inteiro i:

- é um ponteiro *p* (contém endereço)
- deslocado de i * sizeof(tipo apontado pelo ponteiro)



Um vector é um ponteiro para a base acrescido de um deslocamento

A soma de um ponteiro p com um inteiro i:

- é um ponteiro *p* (contém endereço)
- deslocado de i * sizeof(tipo apontado pelo ponteiro)

$$v[i] \equiv v+i$$



Um vector é um ponteiro para a base acrescido de um deslocamento

A soma de um ponteiro p com um inteiro i:

- é um ponteiro *p* (contém endereço)
- deslocado de i * sizeof(tipo apontado pelo ponteiro)

```
v[i] = v+i

LOCAL -4; v
LDINT
LOCAL -8; i
LDINT
INT 4; 8 bytes if double, 4 otherwise
MUL
ADD
```

Alocação de memória



Alocação de memória

```
int *i = [3];
```

Alocação de memória



Alocação de memória

```
int *i = [3];
```

...

ALLOC SP

Alocação de memória

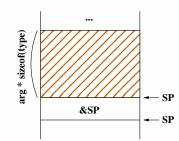


Alocação de memória

$$int *i = [3];$$

•••

ALLOC SP



Exercício



```
int *click(int *x, int dim) {
  int *res, i;
  for ( i = dim -2, res = x+dim -1; i >= 0; i--)
    if (x[i] > *res) res = &x[i];
  return res;
}
```

Exercício



```
int *click(int *x, int dim) {
  int *res, i;
   for (i = \dim -2, res = x + \dim -1; i >= 0; i --)
     if (x[i] > *res) res = &x[i];
   return res;
TEXT
ALIGN
GLOBAL click, FUNC
LABEL click; x@+8 dim@+12
ENTER 8
         ; res@-4 i@-8
LOCAL 12
I.DTNT
INT 2
SUB
DUP32
LOCAL -8
STINT
TRASH 4
LOCAL 8
LDINT
LOCAL 12
LDINT
INT 1
SUB
INT 4
MUL
ADD
DUP32
LOCAL -4
STINT
```

TRASH 4



```
int *click(int *x, int dim) {
  int *res, i;
   for (i = \dim -2, res = x + \dim -1; i >= 0; i --)
     if(x[i] > *res) res = &x[i];
   return res;
TEXT
ALIGN
GLOBAL click, FUNC
                                         AT.TGN
LABEL click; x@+8 dim@+12
                                         LABEL forcond
ENTER 8
         ; res@-4 i@-8
                                         LOCAL -8
                                         LDINT
LOCAL 12
                                         INT 0
I.DTNT
                                         GF.
INT 2
                                         JZ forend
SUB
DUP32
LOCAL -8
STINT
TRASH 4
LOCAL 8
LDINT
LOCAL 12
LDINT
INT 1
SUB
INT 4
MUL
ADD
DUP32
LOCAL -4
STINT
TRASH 4
```



```
int *click(int *x, int dim) {
  int *res, i;
   for (i = \dim -2, res = x + \dim -1; i >= 0; i --)
     if (x[i] > *res) res = &x[i];
   return res;
TEXT
ALIGN
GLOBAL click, FUNC
                                          AT.TGN
LABEL click; x@+8 dim@+12
                                          LABEL forcond
ENTER 8
          ; res@-4 i@-8
                                          LOCAL -8
                                          LDINT
LOCAL 12
                                          INT 0
I.DTNT
                                          GF.
INT 2
                                          JZ forend
SUB
DUP32
LOCAL -8
STINT
                                          LOCAL +8
TRASH 4
                                          LDINT
                                          LOCAL -8
LOCAL 8
                                          LDINT
LDINT
                                          TNT 4
LOCAL 12
                                          MUL
LDINT
                                          ADD
INT 1
                                          LDINT; x[i] = *(x+i)
SUB
                                          LOCAL -4
TNT 4
                                          LDINT ; res
MUL
                                          LDINT ; * res
ADD
                                          GT
DUP32
                                          JZ ifend
LOCAL -4
STINT
```

TRASH 4



```
int *click(int *x, int dim) {
  int *res, i;
   for (i = \dim -2, res = x + \dim -1; i >= 0; i --)
     if (x[i] > *res) res = &x[i];
   return res;
TEXT
                                                                    T.OCAT. 8
ALIGN
                                                                    I.DTNT
GLOBAL click, FUNC
                                                                    LOCAL -8
                                           AT.TGN
LABEL click; x@+8 dim@+12
                                                                    LDINT
                                           LABEL forcond
ENTER 8
          ; res@-4 i@-8
                                                                    TNT 4
                                           LOCAL -8
                                                                    MUL
                                           LDINT
LOCAL 12
                                                                    ADD ; x+i=&x[i]
                                           INT 0
I.DTNT
                                                                    DUP32
                                           GF.
INT 2
                                                                    LOCAL -4
                                           JZ forend
SUB
                                                                    STINT
DUP32
                                                                    TRASH 4
LOCAL -8
STINT
                                           LOCAL +8
TRASH 4
                                           LDINT
                                           LOCAL -8
T.OCAT. 8
                                           LDINT
LDINT
                                           TNT 4
LOCAL 12
                                           MUL
LDINT
                                           ADD
INT 1
                                           LDINT; x[i] = *(x+i)
SUB
                                           LOCAL -4
TNT 4
                                           LDINT ; res
MUL
                                           LDINT ; * res
ADD
                                           GT
DUP32
                                           JZ ifend
LOCAL -4
STINT
```

TRASH 4



```
int *click(int *x, int dim) {
  int *res, i;
   for (i = \dim -2, res = x + \dim -1; i >= 0; i --)
      if (x[i] > *res) res = &x[i];
   return res;
TEXT
                                                                     T.OCAT. 8
ALTGN
                                                                     I.DTNT
GLOBAL click, FUNC
                                                                     LOCAL -8
                                            AT.TGN
LABEL click; x@+8 dim@+12
                                                                     LDINT
                                            LABEL forcond
ENTER 8
          ; res@-4 i@-8
                                                                     TNT 4
                                            LOCAL -8
                                                                     MUL
                                            I.DTNT
LOCAL 12
                                                                      ADD ; x+i=&x[i]
                                            INT 0
I.DTNT
                                                                     DUP32
                                            GF.
INT 2
                                                                     LOCAL -4
                                            JZ forend
SUB
                                                                     STINT
DUP32
                                                                     TRASH 4
LOCAL -8
STINT
                                            LOCAL +8
TRASH 4
                                            LDINT
                                                                     ALIGN
                                            LOCAL -8
                                                                     LABEL ifend
T.OCAT. 8
                                            LDINT
LDINT
                                            TNT 4
                                                                     AT.TGN
LOCAL 12
                                            MUL
                                                                     LABEL forincr
LDINT
                                            ADD
                                                                     LOCAL -8
INT 1
                                            LDINT; x[i] = *(x+i)
                                                                     LDINT
SUB
                                            LOCAL -4
                                                                     DUP32
TNT 4
                                            LDINT ; res
                                                                     INT 1
MUL
                                            LDINT ; * res
                                                                      SUB
ADD
                                            GT
                                                                     LOCAL -8
DUP32
                                           JZ ifend
                                                                     STINT
LOCAL -4
                                                                     TRASH 4
STINT
                                                                     JMP forcond
TRASH 4
```



```
int *click(int *x, int dim) {
  int *res, i;
   for (i = \dim -2, res = x + \dim -1; i >= 0; i --)
      if (x[i] > *res) res = &x[i];
   return res;
TEXT
                                                                      T.OCAT. 8
ALTGN
                                                                      I.DTNT
GLOBAL click, FUNC
                                                                      LOCAL -8
                                            AT.TGN
LABEL click; x@+8 dim@+12
                                                                      LDINT
                                            LABEL forcond
ENTER 8
           ; res@-4 i@-8
                                                                      TNT 4
                                            LOCAL -8
                                                                      MUL
                                            I.DTNT
LOCAL 12
                                                                      ADD ; x+i=&x[i]
                                            INT 0
I.DTNT
                                                                      DUP32
                                            GF.
INT 2
                                                                      LOCAL -4
                                            JZ forend
SUB
                                                                      STINT
DUP32
                                                                      TRASH 4
LOCAL -8
STINT
                                            LOCAL +8
TRASH 4
                                            LDINT
                                                                      ALIGN
                                            LOCAL -8
                                                                      LABEL ifend
T.OCAT. 8
                                            LDINT
LDINT
                                            TNT 4
                                                                      AT.TGN
LOCAL 12
                                            MUL
                                                                      LABEL forincr
I.DTNT
                                            ADD
                                                                      LOCAL -8
INT 1
                                            LDINT; x[i] = *(x+i)
                                                                      LDINT
SUB
                                            LOCAL -4
                                                                      DUP32
TNT 4
                                            LDINT ; res
                                                                      INT 1
MUL
                                            LDINT ; * res
                                                                      SUB
ADD
                                            GT
                                                                      LOCAL -8
DUP32
                                            JZ ifend
                                                                      STINT
LOCAL -4
                                                                      TRASH 4
STINT
                                                                      JMP forcond
TRASH 4
```

ALIGN LABEL forend LOCAL -4 LDINT STEVAL32

LEAVE RET

Arranque do programa



Funções:

- Normalmente, as funções nomeadas existem no contexto global
- Em L22:
 - Apenas existe uma função "nomeada": o main
 - O início e fim do main está delimitado pelas keywords begin e end
 - O resto de funções não são nomeadas e podem existir funções dentro de funções (Atenção: TEXT)

Arranque do programa



Funções:

- Normalmente, as funções nomeadas existem no contexto global
- Em L22:
 - Apenas existe uma função "nomeada": o main
 - O início e fim do main está delimitado pelas keywords begin e end
 - O resto de funções não são nomeadas e podem existir funções dentro de funções (Atenção: TEXT)

```
Ponto de entrada: Label _main
TEXT
ALIGN
GLOBAL _main, FUNC
LABEL _main
ENTER ...
```



Conversão directa de código POSTFIX para código assembly:

compilador pf2asm
 (https://web.tecnico.ulisboa.pt/~david.matos/w/pt/images/d/d9/Pf2asm-202206081024.tar.bz2)



Conversão directa de código POSTFIX para código assembly:

compilador pf2asm
 (https://web.tecnico.ulisboa.pt/~david.matos/w/pt/images/d/d9/Pf2asm-202206081024.tar.bz2)

O projecto gera directamente código assembly!

- Visitor postfix_writer, chama métodos cdk::basic_postfix_emitter cdk::postfix_ix86_emitter
- Opção –g para modo debug



Conversão directa de código POSTFIX para código assembly:

compilador pf2asm
 (https://web.tecnico.ulisboa.pt/~david.matos/w/pt/images/d/d9/Pf2asm-202206081024.tar.bz2)

O projecto gera directamente código assembly!

- Visitor postfix_writer, chama métodos cdk::basic_postfix_emitter cdk::postfix_ix86_emitter
- Opção –g para modo debug

Vantagens do pf2asm:

- Geração directa de protótipos em código POSTFIX
- Permite a realização de testes



Exemplo - A-01-1-N-ok.l22

begin writeln "string" return 0 end



Exemplo - A-01-1-N-ok.l22

```
begin
  writeln "string"
  return O
end
TEXT
ALTGN
                     CALL println
GLOBAL _main, FUNC
                     INT O
LABEL main
                     STFVAL32
ENTER O
RODATA
                     JMP _L1
                     LABEL _L1
ALIGN
LABEL _L2
                     LEAVE
SSTRING "string"
                     RET
                     EXTERN println
TEXT
                     EXTERN prints
ADDR L2
CALL prints
```

TRASH 4



Exemplo - A-01-1-N-ok.l22

```
begin
  writeln "string"
  return O
end
                                                     segment .text
TEXT
                                                     align
ALTGN
                                                     global
                                                             _main:function
                                                                                           call
                                                                                                   println
                      CALL println
GLOBAL main, FUNC
                                                     _main:
                                                                                                   dword 0
                                                                                           push
                      INT O
                                                                     ebp
LABEL main
                                                             push
                                                                                           pop
                                                                                                    eax
ENTER O
                      STFVAL32
                                                             mov
                                                                     ebp, esp
                                                                                           jmp
                      JMP L1
                                                                     esp, 0
RODATA
                                                             sub
                                                                                   dword I.1
                      LABEL L1
                                                     segment .rodata
ALTGN.
                                                                                   L1:
LABEL _L2
                      LEAVE
                                                     align
                                                             4
                                                                                           leave
                                                     _L2:
SSTRING "string"
                      RET
                                                                                           ret.
                      EXTERN println
                                                                      "string", 0
TEXT
                                                             db
                                                                                   extern println
                      EXTERN prints
                                                     segment .text
ADDR L2
                                                                                   extern prints
CALL prints
                                                             push
                                                                     dword $_L2
                                                             call
                                                                     prints
TRASH 4
                                                                     esp, 4
                                                             add
```

Comando: $pf2asm file.pf \rightarrow file.asm$

Geração de código binário



Yasm Modular Assembler

- Assembler para a arquitectura Intel x86
- Permite múltiplos formatos de output (ELF, win32, ...)
 https://en.wikipedia.org/wiki/Executable_and_Linkable_Format

Comando: yasm -felf -o file.o file.asm \rightarrow file.o

Geração de um executável



Linker:

- Ligação dos módulos binários e bibliotecas num executável
- Linux: GNU 1d

Geração de um executável



Linker:

- Ligação dos módulos binários e bibliotecas num executável
- Linux: GNU 1d

A biblioteca RTS fornece funções para interação:

- com o ambiente de execução
- com o programador

Manual da RTS:

```
https:
```

 $// web.tecnico.ulisboa.pt/~david.matos/w/pt/index.php/Manual_da_RTS$

Geração de um executável



Linker:

- Ligação dos módulos binários e bibliotecas num executável
- Linux: GNU 1d

A biblioteca RTS fornece funções para interação:

- com o ambiente de execução
- com o programador

Manual da RTS:

```
https:
```

 $//{\tt web.tecnico.ulisboa.pt/~david.matos/w/pt/index.php/Manual_da_RTS}$

Comando:

```
ld -m elf_i386 -o file file.o -L$HOME/compiladores/root/usr/lib -lrts \rightarrow file
```





```
static double v[] = \{ 1.41, 2.71, 3.14 \}; int f(double \ v[], int \ s, \ double \ d) \ \{ for (int \ i = 0; \ i < s; \ i++) \\ if (v[i] == d) \ return \ i; return \ -1; \} int main() \{ return \ f(v, 3, 3.14); \}
```

```
DATA
ALIGN
LABEL v
SDOUBLE 1.41
SDOUBLE 2.71
SDOUBLE 3.14
```



```
static double v[] = \{ 1.41, 2.71, 3.14 \};
int f(double v[], int s, double d) {
  for (int i = 0; i < s; i++)
    if (v[i] = d) return i;
  return -1:
int main() {
  return f(v, 3, 3.14);
DATA
ALTGN.
LABEL v
SDOUBLE 1.41
SDOUBLE 2.71
SDOUBLE 3.14
TEXT
ALIGN
GLOBAL f, FUNC
LABEL f
ENTER 4
; i@-4 v@8 s@+12 d@+16
```



```
static double v[] = \{ 1.41, 2.71, 3.14 \};
int f(double v[], int s, double d) {
  for (int i = 0; i < s; i++)
    if (v[i] = d) return i;
  return -1:
int main() {
  return f(v, 3, 3.14);
                       INT O
                       LOCAL -4
                       STINT
DATA
                       ALTGN
ALTGN.
                       LABEL forcond
LABEL v
                       LOCAL -4
SDOUBLE 1.41
                       LDINT
SDOUBLE 2.71
                       LOCAL 12
SDOUBLE 3.14
                       LDINT
                       I.T
                       JZ forend
TEXT
ALIGN
                       LOCAL +8
GLOBAL f, FUNC
                       LDINT
LABEL f
                       LOCAL -4
ENTER 4
                       LDINT
; i@-4 v@8 s@+12 d@+16
                       TNT 8
                       MUL
                       ADD
                       LDDOUBLE
```



```
static double v[] = \{ 1.41, 2.71, 3.14 \};
int f(double v[], int s, double d) {
  for (int i = 0; i < s; i++)
    if (v[i] = d) return i;
  return -1:
int main() {
  return f(v, 3, 3.14);
                       INT O
                       LOCAL -4
                       STINT
DATA
                       ALTGN.
                                       LOCAL 16
ALTGN.
                       LABEL forcond
                                       LDDOUBLE
LABEL v
                       LOCAL -4
                                       DCMP
SDOUBLE 1.41
                       LDINT
                                       TNT O
SDOUBLE 2.71
                       LOCAL 12
                                       EQ
SDOUBLE 3.14
                       LDINT
                                       JZ ifend
                       I.T
                       JZ forend
                                       LOCAL -4
TEXT
                                       LDINT
ALIGN
                       LOCAL +8
                                       STFVAL32
GLOBAL f, FUNC
                       LDINT
                                       LEAVE
LABEL f
                       LOCAL -4
                                       RET
ENTER 4
                       LDINT
                                       ALIGN
; i@-4 v@8 s@+12 d@+16
                       TNT 8
                                       LABEL ifend
                       MUL
                       ADD
                       LDDOUBLE
```



```
static double v[] = \{ 1.41, 2.71, 3.14 \};
int f(double v[], int s, double d) {
  for (int i = 0; i < s; i++)
    if (v[i] = d) return i;
  return -1:
int main() {
  return f(v, 3, 3.14);
                       INT O
                                                        ALIGN
                       LOCAL -4
                                                        LAREL forincr
                       STINT
                                                        I.OCAI. -4
DATA
                       ALTGN
                                       LOCAL 16
                                                        LDINT
ALTGN.
                       LABEL forcond
                                       LDDOUBLE
                                                        DUP32
LABEL v
                       LOCAL -4
                                       DCMP
                                                        TNT 1
SDOUBLE 1.41
                       LDINT
                                       TNT O
                                                        ADD
SDOUBLE 2.71
                       LOCAL 12
                                        EQ
                                                        LOCAL -4
SDOUBLE 3.14
                       LDINT
                                        JZ ifend
                                                        STINT
                       I.T
                                                        TRASH 4
                       JZ forend
                                        LOCAL -4
                                                        JMP forcond
TEXT
                                       LDINT
ALIGN
                       LOCAL +8
                                        STFVAL32
GLOBAL f, FUNC
                       LDINT
                                        LEAVE
LABEL f
                       LOCAL -4
                                        RET
ENTER 4
                       LDINT
                                        ALIGN
; i@-4 v@8 s@+12 d@+16
                       TNT 8
                                        LABEL ifend
                       MUL
                       ADD
```

LDDOUBLE



```
static double v[] = \{ 1.41, 2.71, 3.14 \};
int f(double v[], int s, double d) {
  for (int i = 0; i < s; i++)
    if (v[i] = d) return i;
  return -1:
int main() {
  return f(v, 3, 3.14);
                       INT O
                                                        ALIGN
                       LOCAL -4
                                                        LAREL forincr
                       STINT
                                                        I.OCAI. -4
DATA
                       ALTGN
                                        LOCAL 16
                                                        LDINT
ALTGN
                       LABEL forcond
                                       LDDOUBLE
                                                        DUP32
LABEL v
                       LOCAL -4
                                       DCMP
                                                        TNT 1
SDOUBLE 1.41
                       LDINT
                                       TNT O
                                                        ADD
SDOUBLE 2.71
                       LOCAL 12
                                        EQ
                                                        LOCAL -4
SDOUBLE 3.14
                       LDINT
                                        JZ ifend
                                                        STINT
                       I.T
                                                        TRASH 4
                       JZ forend
                                        LOCAL -4
                                                        JMP forcond
TEXT
                                        LDINT
ALIGN
                       LOCAL +8
                                        STFVAL32
GLOBAL f, FUNC
                       LDINT
                                        LEAVE
                                                        ALIGN
LABEL f
                                                        LABEL forend
                       LOCAL -4
                                        RET
ENTER 4
                       LDINT
                                        ALIGN
; i@-4 v@8 s@+12 d@+16
                                                        INT -1
                       TNT 8
                                        LABEL ifend
                                                        STFVAL32
                       MUL
                                                        LEAVE
                       ADD
                       LDDOUBLE
                                                        RET
```



```
static double v[] = \{ 1.41, 2.71, 3.14 \};
int f(double v[], int s, double d) {
  for (int i = 0; i < s; i++)
    if (v[i] == d) return i:
  return -1:
int main() {
  return f(v, 3, 3.14);
                        INT O
                                                         ALIGN
                        LOCAL -4
                                                         LAREL forincr
                                                                                    TEXT
                        STINT
                                                                                    ALIGN
                                                         I.OCAI. -4
DATA
                        ALTGN.
                                        LOCAL 16
                                                                                    GLOBAL _main, FUNC
                                                         LDINT
ALTGN
                        LABEL forcond
                                        LDDOUBLE
                                                                                    LABEL main
                                                         DUP32
LABEL v
                        LOCAL -4
                                        DCMP
                                                                                    ENTER O
                                                         TNT 1
SDOUBLE 1.41
                        LDINT
                                        TNT O
                                                         ADD
SDOUBLE 2.71
                        LOCAL 12
                                        EQ
                                                                                    DOUBLE 3.14
                                                         LOCAL -4
SDOUBLE 3.14
                        LDINT
                                        JZ ifend
                                                         STINT
                                                                                    INT 3
                        I.T
                                                         TRASH 4
                                                                                    ADDR. v
                        JZ forend
                                        LOCAL -4
                                                         JMP forcond
TEXT
                                        LDINT
                                                                                    CALL f
ALIGN
                        LOCAL +8
                                        STFVAL32
                                                                                    TRASH 16
GLOBAL f, FUNC
                        LDINT
                                        LEAVE
                                                         ALIGN
LABEL f
                                                         LABEL forend
                        LOCAL -4
                                        RET
                                                                                    LDFVAL32
ENTER 4
                        LDINT
                                        ALIGN
; i@-4 v@8 s@+12 d@+16
                                                         INT -1
                        TNT 8
                                        LABEL ifend
                                                                                    STFVAL32
                                                         STFVAL32
                        MUL
                                                                                    LEAVE.
                                                         LEAVE
                        ADD
                                                                                    RET
                        LDDOUBLE
                                                         RET
```

Questões?



Dúvidas?