Linguagens de Programação

Prof. Miguel Elias Mitre Campista

http://www.gta.ufrj.br/~miguel

Parte I

Ferramentas de Programação

Introdução

- GCC (GNU C Compiler)
  - Autor: Richard Stallman
    - Fundador do Projeto GNU
- Projeto GNU, iniciado em 1984
  - Objetivo: criação de um sistema operacional totalmente livre baseado em UNIX
    - Em 1984, não havia compiladores gratuitos
      - Projeto GNU necessitou criar o seu próprio

Introdução

- Primeira versão do GCC surgiu em 1987
  - Primeiro compilador C ANSI portável lançado como software livre
    - Pode ser executado em diferentes plataformas e produz saídas para vários tipos de processadores
- Versão revisada em 1992
  - Adiciona a possibilidade de compilação de C++
- Com o passar dos anos...
  - GCC foi estendido para dar suporte a outras linguagens:
    - Fortran, ADA, Java e Objective-C

O acrônimo GCC passou a ser referenciado como *GNU Compiler Collection*

Introdução

- GNU Fortran, por exemplo, é baseado no GCC
  - Logo, compartilham muitas características

Programa: hello.f

```
program hello

    print *,"Hello World!"

end program hello
```

Compilação: gfortran -o h hello.f

```
shell$> ./h
 Hello World!
```

Compilação

- Processo de conversão de um programa em código fonte textual em código de máquina
  - Código em C/C++ ➔ Sequência de 1's e 0's

Programa em C: `hello.c`

Compilação: `gcc -Wall hello.c -o h`

Programa em C++: `hello.cpp`

Compilação: `g++ -Wall hello.cpp -o h`

Compilação

- Opção: -Wall (***Warning all***)
  - Representa uma importante ajuda para detecção de problemas em C/C++
  - Deve ser **sempre** usada
- Formato das mensagens:
  - **arquivo:número_da_linha:mensagem**

```c
#include "stdio.h"

int main () {
        printf ("Dois mais dois é %f\n", 4);
        return 0;
}
```

Erro de execução!!!

```
miguel@pegasus-linux:~$ ./gcc-ex1
Dois mais dois é -1.993767
```

```
miguel@pegasus-linux:~$ gcc -Wall gcc-ex1.c -o gcc-ex1
gcc-ex1.c: In function ‘main’:
gcc-ex1.c:4: warning: format ‘%f’ expects type ‘double’, but argument 2 has type ‘int’
```

Compilação de Múltiplos Arquivos

- Programas podem ser divididos em múltiplos arquivos
  - Simplifica a edição e a compreensão

gcc-ex2.c

```c
#include "gcc-hello-ex2.h"

int main () {
        hello ("WORLD");
        return 0;
}
```

gcc-hello-ex2.h

```c
void hello (const char *);
```

gcc-hello-ex2.c

```c
#include <stdio.h>
#include "gcc-hello-ex2.h"

void hello (const char * nome) {
        printf ("Hello, %s!\n", nome);
}
```

```
miguel@pegasus-linux:~$ gcc -Wall gcc-hello-ex2.c gcc-ex2.c -o gcc-ex2
miguel@pegasus-linux:~$ ./gcc-ex2
Hello, WORLD!
```

Compilação de Múltiplos Arquivos

- Compilação dos arquivos de maneira independente
  - Opção: -c (Compila arquivo em arquivo objeto)
    - Por padrão, nome do arquivo objeto é o mesmo do arquivo fonte, com extensão diferente

```
miguel@pegasus-linux:~$ ls gcc-*ex2.*
gcc-ex2.c  gcc-hello-ex2.c  gcc-hello-ex2.h
miguel@pegasus-linux:~$ gcc -Wall -c gcc-ex2.c
miguel@pegasus-linux:~$ ls gcc-*ex2.*
gcc-ex2.c  gcc-ex2.o  gcc-hello-ex2.c  gcc-hello-ex2.h
miguel@pegasus-linux:~$ gcc -Wall -c gcc-hello-ex2.c
miguel@pegasus-linux:~$ ls gcc-*ex2.*
gcc-ex2.c  gcc-ex2.o  gcc-hello-ex2.c  gcc-hello-ex2.h  gcc-hello-ex2.o
```

Arquivos objetos possuem referências a funções externas, mas não possuem ainda o endereço de memória correspondente que permanece indefinido

Compilação de Múltiplos Arquivos

- Criação de arquivos executáveis
  - Arquivos objetos são ligados (*linked*)
    - GCC usa o programa ligador `ld`
  - Não há necessidade de usar o -Wall
    - Opção já deve ter sido usada para compilar os arquivos fonte e nada mais é alertado

```
miguel@pegasus-linux:~$ gcc gcc-hello-ex2.o gcc-ex2.o -o gcc-ex2
miguel@pegasus-linux:~$ ./gcc-ex2
Hello, WORLD!
```

Processo de ligação preenche os endereços para as funções externas que estavam indefinidos

Compilação de Múltiplos Arquivos

- Recompilação de arquivos fonte
  - Apenas o arquivo fonte alterado precisa ser recompilado e o programa deve ser religado
    - Simplifica a recompilação em casos de programas muito grandes e com muitos arquivos
    - Dispensa a necessidade dos usuários terem todos os arquivos fonte de um mesmo programa

```
#include "gcc-hello-ex2.h"

int main () {
        hello ("TURMA");
        return 0;
}
```

```
miguel@pegasus-linux:~$ gcc -c gcc-ex2.c
miguel@pegasus-linux:~$ gcc gcc-hello-ex2.o gcc-ex2.o -o gcc-ex2
miguel@pegasus-linux:~$ ./gcc-ex2
Hello, TURMA!
```

Makefile

- Especifica um conjunto de regras de compilação
  - Alvos (executáveis) e suas dependências (arquivos objeto e arquivos fonte)

```
alvo: dependências
[TAB] comando
```

- Programa **make** lê a descrição de um projeto no Makefile
  - Para cada alvo, o **make** verifica quando as dependências foram modificadas pela última vez para determinar se o alvo precisa ser reconstruído

Makefile

- Identação da linha de comando é feita com um TAB
- Começa sempre com o primeiro alvo
  - Chamado de objetivo padrão (*default goal*)
- Regras implícitas
  - Regras definidas por padrão
    - Exs.: Arquivos ".o" são obtidos de arquivos ".c" através da compilação

      Arquivos executáveis são obtidos pela ligação de arquivos ".o"
  - Definidas em termos de variáveis do **make**
    - Exs.: CC (compilador C)

      CFLAGS (opções de compilação em programas em C)

Atribuição é do tipo `VARIÁVEL=VALOR`

Makefile

```
CC=gcc
CFLAGS=-Wall

gcc-ex2:
	$(CC) $(CFLAGS) gcc-ex2.c gcc-hello-ex2.c -o gcc-ex2


clean:
	rm -vf gcc-ex2 *.o
```

Se fosse C++, bastaria substituir as variáveis CC para CPP e CFLAGS para CPPFLAGS, além de substituir o compilador para g++.

```
miguel@pegasus-linux:~$ make
gcc -Wall gcc-ex2.c gcc-hello-ex2.c -o gcc-ex2
miguel@pegasus-linux:~$ ./gcc-ex2
Hello, TURMA!
miguel@pegasus-linux:~$ make clean
rm -vf gcc-ex2 *.o
`gcc-ex2' removido
```

Makefile

```
CC=gcc
CFLAGS=-Wall

gcc-ex2: gcc-ex2.o gcc-hello-ex2.o
	$(CC) gcc-ex2.o gcc-hello-ex2.o -o gcc-ex2

gcc-ex2.o: gcc-ex2.c
	$(CC) $(CFLAGS) -c gcc-ex2.c

gcc-hello-ex2.o: gcc-hello-ex2.c
	$(CC) $(CFLAGS) -c gcc-hello-ex2.c

clean:
	rm -vf gcc-ex2 *.o
```

Com dependências

```
miguel@pegasus-linux:~$ make
gcc -Wall -c gcc-ex2.c
gcc -Wall -c gcc-hello-ex2.c
gcc gcc-ex2.o gcc-hello-ex2.o -o gcc-ex2
miguel@pegasus-linux:~$ ./gcc-ex2
Hello, TURMA!
miguel@pegasus-linux:~$ make clean
rm -vf gcc-ex2 *.o
`gcc-ex2' removido
`gcc-ex2.o' removido
`gcc-hello-ex2.o' removido
```

Makefile

```
CC=gcc
CFLAGS=-Wall

gcc-ex2: gcc-ex2.o gcc-hello-ex2.o

clean:
	rm -vf gcc-ex2 *.o
```

Com regras implícitas

```
miguel@pegasus-linux:~$ make
gcc -Wall   -c -o gcc-ex2.o gcc-ex2.c
gcc -Wall   -c -o gcc-hello-ex2.o gcc-hello-ex2.c
gcc   gcc-ex2.o gcc-hello-ex2.o   -o gcc-ex2
miguel@pegasus-linux:~$ ./gcc-ex2
Hello, TURMA!
miguel@pegasus-linux:~$ make clean
rm -vf gcc-ex2 *.o
`gcc-ex2' removido
`gcc-ex2.o' removido
`gcc-hello-ex2.o' removido
```

Makefile

```
CC=gcc
CFLAGS=-Wall

all: gcc-ex2.o gcc-hello-ex2.o
	$(CC) gcc-ex2.o gcc-hello-ex2.o -o gcc-ex2

clean:
	rm -vf gcc-ex2 *.o
```

Com alvo "all"

```
miguel@pegasus-linux:~$ make
gcc -Wall   -c -o gcc-ex2.o gcc-ex2.c
gcc -Wall   -c -o gcc-hello-ex2.o gcc-hello-ex2.c
gcc gcc-ex2.o gcc-hello-ex2.o -o gcc-ex2
miguel@pegasus-linux:~$ ./gcc-ex2
Hello, TURMA!
miguel@pegasus-linux:~$ make clean
rm -vf gcc-ex2 *.o
`gcc-ex2' removido
`gcc-ex2.o' removido
`gcc-hello-ex2.o' removido
```

Makefile

```
CC=gcc
CFLAGS=-Wall

OBJECTS = gcc-ex2.o gcc-hello-ex2.o

all: $(OBJECTS)
	$(CC) $(OBJECTS) -o gcc-ex2

gcc-ex2.o: gcc-ex2.c
	$(CC) $(CFLAGS) -c gcc-ex2.c

gcc-hello-ex2.o: gcc-hello-ex2.c
	$(CC) $(CFLAGS) -c gcc-hello-ex2.c

clean:
	rm -vf gcc-ex2 *.o
```

Com macros

```
miguel@pegasus-linux:~$ make
gcc -Wall -c gcc-ex2.c
gcc -Wall -c gcc-hello-ex2.c
gcc gcc-ex2.o gcc-hello-ex2.o -o gcc-ex2
miguel@pegasus-linux:~$ ./gcc-ex2
Hello, TURMA!
miguel@pegasus-linux:~$ make clean
rm -vf gcc-ex2 *.o
`gcc-ex2' removido
`gcc-ex2.o' removido
`gcc-hello-ex2.o' removido
```

Makefile

- Macros especiais
  - $@
    - Nome completo do alvo atual
  - $?
    - Lista de arquivos desatualizados para dependência atual
  - $<
    - Arquivo fonte da única dependência atual

Makefile

Com macros especiais

```
CC=gcc
CFLAGS=-Wall

OBJECTS = gcc-ex2.o gcc-hello-ex2.o

PROGRAM = gcc-ex2

all: $(PROGRAM)

$(PROGRAM): $(OBJECTS)
	$(CC) $(OBJECTS) -o $@

.c.o:
	$(CC) $(CFLAGS) -c $<

clean:
	rm -vf gcc-ex2 *.o
```

```
miguel@pegasus-linux:~$ make
gcc -Wall -c gcc-ex2.c
gcc -Wall -c gcc-hello-ex2.c
gcc gcc-ex2.o gcc-hello-ex2.o -o gcc-ex2
miguel@pegasus-linux:~$ ./gcc-ex2
Hello, TURMA!
miguel@pegasus-linux:~$ make clean
rm -vf gcc-ex2 *.o
`gcc-ex2' removido
`gcc-ex2.o' removido
`gcc-hello-ex2.o' removido
```

Bibliotecas Externas

- Conjuntos de arquivos objetos pré-compilados
  - Podem ser ligados aos programas
- Provêem funções ao sistema
  - Ex.: Função **`sqrt`** da biblioteca matemática do *C*
- Encontradas em **`/usr/lib`** e **`/lib`**
  - Ex.: Biblioteca matemática: **`/usr/lib/libm.a`**

    Biblioteca padrão *C*: **`/usr/lib/libc.a`**, contém funções como o **`printf`** e é ligada a todos os programas por padrão

Bibliotecas Externas

- Arquivo **math.h**
  - Contém os protótipos das funções da biblioteca matemática em /usr/lib/libm.a
    - No diretório padrão **/usr/include/**

```c
#include <math.h>
#include <stdio.h>

int main () {
        double x = sqrt (2.0);
        printf ("A raiz quadrada de 2.0 eh %f\n", x);
        return 0;
}
```

-l*NOME* é um atalho para /usr/lib/lib*NOME*.a

```
miguel@pegasus-linux:~$ gcc -Wall gcc-ex3.c /usr/lib/libm.a -o gcc-ex3
miguel@pegasus-linux:~$ ./gcc-ex3
A raiz quadrada de 2.0 eh 1.414214
miguel@pegasus-linux:~$ gcc -Wall gcc-ex3.c -lm -o gcc-ex3
miguel@pegasus-linux:~$ ./gcc-ex3
A raiz quadrada de 2.0 eh 1.414214
```

Bibliotecas Externas

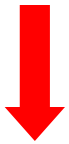
- Definição das funções deve aparecer primeiro que as funções propriamente ditas
  - Logo, as bibliotecas devem aparecer depois dos arquivos fontes ou arquivos objetos

```
gcc -Wall -lm gcc-ex3.c -o gcc-ex3
```

```
gcc -Wall gcc-ex3.c -lm -o gcc-ex3
```

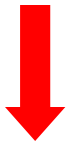
Opções de Compilação

- Diretórios padrão de arquivos de cabeçalho
  - `/usr/local/include`
  - `/usr/include`



Ordem de busca (precedência)

- Diretórios padrão de bibliotecas
  - `/usr/local/lib`
  - `/usr/lib`



Ordem de busca (precedência)

Se um arquivo de cabeçalho for encontrado em `/usr/local/include`, o compilador não busca mais em `/usr/include`. O mesmo para as bibliotecas

Opções de Compilação

- Opção –I:
  - Adiciona novo diretório para o início do caminho de busca por arquivos de cabeçalho
- Opção –L:
  - Adiciona novo diretório para o início do caminho de busca por bibliotecas

Assumindo que o programa utilize um sistema de banco de dados (GNU Database Management - GDBM):

```
gcc -Wall -I/opt/gdbm-1.8.3/include -L/opt/gdbm-1.8.3/lib
gcc-ex4.c -lgdbm
```

Variáveis de Ambiente

- Ao invés de sempre adicionar os diretórios, pode-se definir variáveis de ambiente
  - Variáveis podem ser definidas em arquivos do shell

Arquivos de cabeçalho:

```
C_INCLUDE_PATH=/opt/gdbm-1.8.3/include
export C_INCLUDE_PATH
```

Se fosse C++...

```
CPLUS_INCLUDE_PATH=/opt/gdbm-1.8.3/include
export CPLUS_INCLUDE_PATH
```

Bibliotecas:

```
LIBRARY_PATH=/opt/gdbm-1.8.3/lib
export LIBRARY_PATH
```

Assim...

```
gcc -Wall gcc-ex4.c -lgdbm -o gcc-ex4
```

Variáveis de Ambiente

- Para inserir múltiplos diretórios...

```
gcc -Wall -I. -I/opt/gdbm-1.3.8/include -I/net/include
-L. -L/opt/gdbm-1.3.8/lib -L/net/lib gcc-ex4.c -lgdbm
-lnet -o gcc-ex4
```

- Através das variáveis de ambiente...

```
C_INCLUDE_PATH=.:/opt/gdbm-1.3.8/include:/net/include
LIBRARY_PATH=.:/opt/gdbm-1.3.8/lib:/net/lib

gcc -Wall gcc-ex4.c -lgdbm -o gcc-ex4
```

Bibliotecas Estáticas

- Bibliotecas estáticas (arquivos ".a")
  - Copia da biblioteca o código de máquina das funções externas usadas pelo programa
    - Copia no executável final durante a ligação

Criação de Bibliotecas Estáticas

- Uso da ferramenta: **ar** (*GNU archiver*)
  - Combina uma coleção de arquivos objeto em um único arquivo de biblioteca (arquivo *archive*)
    - Uma biblioteca é uma maneira conveniente de distribuir um número grande de arquivos objetos relacionados em um único arquivo

```
ar <nome_da_biblioteca> <lista_de_arquivos_objeto>
```

Criação de Bibliotecas Estáticas

Arquivo: gcc-hello-ex2.h

```
void hello (const char *);
void bye ();
```

Arquivo: gcc-bye-ex2.c

```
#include <stdio.h>
#include "gcc-hello-ex2.h"

void bye () {
        printf ("Good Bye!\n");
}
```

Arquivo: gcc-hello-ex2.c

```
#include <stdio.h>
#include "gcc-hello-ex2.h"

void hello (const char * nome) {
        printf ("Hello, %s!\n", nome);
}
```

Arquivo: gcc-ex2.c

```
#include "gcc-hello-ex2.h"

int main () {
        hello ("TURMA");
        bye ();
        return 0;
}
```

Criação de Bibliotecas Estáticas

```
miguel@pegasus-linux:~$ gcc -Wall -c gcc-hello-ex2.c -o gcc-hello-ex2.o
miguel@pegasus-linux:~$ gcc -Wall -c gcc-bye-ex2.c -o gcc-bye-ex2.o
miguel@pegasus-linux:~$ ar cr libhello.a gcc-hello-ex2.o gcc-bye-ex2.o
miguel@pegasus-linux:~$ ar t libhello.a
gcc-hello-ex2.o
gcc-bye-ex2.o
miguel@pegasus-linux:~$ gcc -Wall gcc-ex2.c libhello.a -o gcc-ex2
miguel@pegasus-linux:~$ ./gcc-ex2
Hello, TURMA!
Good Bye!
miguel@pegasus-linux:~$ gcc -Wall -L. gcc-ex2.c -lhello -o gcc-ex2
miguel@pegasus-linux:~$ ./gcc-ex2
Hello, TURMA!
Good Bye!
```

Opções do `ar`:
`cr`: create and replace
`t`: table of contents

Bibliotecas Compartilhadas

- Bibliotecas compartilhadas (arquivos ".so")
  - Arquivo executável final contém apenas uma tabela das funções que necessita, ao invés de todo código de máquina
    - Menor arquivo executável final
  - Antes do arquivo executável começar a rodar, o código de máquina das funções externas é copiado para a **memória** pelo Sistema Operacional
    - Transferido do arquivo da biblioteca compartilhada que está em disco
      - Processo chamado de ligação dinâmica (*dynamic linking*)
  - No Windows as bibliotecas *.so são as *.dll

Bibliotecas Compartilhadas

- Bibliotecas compartilhadas (arquivos ".so")
  - Uma vez carregadas em memória...
    - Podem ser compartilhadas por todos os programas em execução que utilizam a mesma biblioteca
  - O gcc busca primeiro a biblioteca compartilhada **lib*NOME*.so** para depois buscar a **lib*NOME*.a**
    - Sempre que a biblioteca **-l*NOME*** é adicionada pelo gcc
  - Variável de ambiente para bibliotecas compartilhadas são carregadas na **LD_LIBRARY_PATH**:
    - **LD_LIBRARY_PATH=/opt/gdbm-1.3.8/lib**

Criação de Bibliotecas Compartilhadas (Dinâmicas)

- Uso da opção **-shared**

```
gcc -shared -o <biblioteca.so> <lista_arquivos_objetos>
```

  - Outra alternativa:
    - Opção **-Wl**: lista opções ao programa ligador separadas por vírgulas

```
gcc -shared -Wl,-soname,<biblioteca.so.versão> -o
<biblioteca.so.versão_completa> <lista_arquivos_objetos>
```

Opções do ligador (`ld`):
`-soname`: quando um executável é ligado a uma biblioteca compartilhada, o ligador dinâmico tenta ligar o executável com a biblioteca de nome passado com a opção `soname` e não com a biblioteca com o nome dado com o `-o`. Isso permite a existência de bibliotecas com nomes e versões diferentes da criada.

Criação de Bibliotecas Compartilhadas (Dinâmicas)

- Dependências de bibliotecas **compartilhadas**: `ldd` (*LD Dependencies*)
  - Verifica quais são e se as bibliotecas compartilhadas necessárias já foram encontradas
    - Caso tenham sido, o caminho da biblioteca é apresentado

```
ldd <arquivo_executável>
```

Criação de Bibliotecas Compartilhadas (Dinâmicas)

Arquivo: gcc-hello-ex2.h

```
void hello (const char *);
void bye ();
```

Arquivo: gcc-bye-ex2.c

```
#include <stdio.h>
#include "gcc-hello-ex2.h"

void bye () {
        printf ("Good Bye!\n");
}
```

Arquivo: gcc-hello-ex2.c

```
#include <stdio.h>
#include "gcc-hello-ex2.h"

void hello (const char * nome) {
        printf ("Hello, %s!\n", nome);
}
```

Arquivo: gcc-ex2.c

```
#include "gcc-hello-ex2.h"

int main () {
        hello ("TURMA");
        bye ();
        return 0;
}
```

Criação de Bibliotecas Compartilhadas (Dinâmicas)

```
miguel@pegasus-linux:~$ gcc -Wall -c gcc-hello-ex2.c -o gcc-hello-ex2.o
miguel@pegasus-linux:~$ gcc -Wall -c gcc-bye-ex2.c -o gcc-bye-ex2.o
miguel@pegasus-linux:~$ gcc -shared -o libhello.so gcc-hello-ex2.o gcc-bye-ex2.o
miguel@pegasus-linux:~$ gcc -Wall -L. gcc-ex2.c -o gcc-ex2 -lhello
miguel@pegasus-linux:~$ ./gcc-ex2
./gcc-ex2: error while loading shared libraries: libhello.so: cannot open shared object file: No suc
h file or directory
miguel@pegasus-linux:~$ ldd gcc-ex2
        linux-gate.so.1 =>  (0xb7fd8000)
        libhello.so => not found
        libc.so.6 => /lib/i686/cmov/libc.so.6 (0xb7e6a000)
        /lib/ld-linux.so.2 (0xb7fd9000)
miguel@pegasus-linux:~$ echo $LD_LIBRARY_PATH
:/home/miguel/simulacao/ns-allinone-2.34/otcl-1.13:/home/miguel/simulacao/ns-allinone-2.34/lib
miguel@pegasus-linux:~$ export LD_LIBRARY_PATH=.:$LD_LIBRARY_PATH
miguel@pegasus-linux:~$ echo $LD_LIBRARY_PATH
.::/home/miguel/simulacao/ns-allinone-2.34/otcl-1.13:/home/miguel/simulacao/ns-allinone-2.34/lib
miguel@pegasus-linux:~$ ldd gcc-ex2
        linux-gate.so.1 =>  (0xb7f06000)
        libhello.so => ./libhello.so (0xb7f02000)
        libc.so.6 => /lib/i686/cmov/libc.so.6 (0xb7d96000)
        /lib/ld-linux.so.2 (0xb7f07000)
miguel@pegasus-linux:~$ ./gcc-ex2
Hello, TURMA!
Good Bye!
```

Uso Forçado das Bibliotecas Estáticas

- Opção –static:
  - Força o uso das bibliotecas estáticas
    - Pode ser vantajoso para evitar a definição de **LD_LIBRARY_PATH**

```
gcc -Wall -static -I/opt/gdbm-1.3.8/include -
L/opt/gdbm-1.3.8/lib gcc-ex4.c -lgdbm -o gcc-ex4
```

- Outra maneira de forçar o uso de uma determinada biblioteca é colocando o caminho completo...

```
gcc -Wall -I/opt/gdbm-1.3.8/include gcc-ex4.c
/opt/gdbm-1.3.8/lib/libgdbm.a -o gcc-ex4
```

Comparação entre Bibliotecas Compartilhadas e Estáticas

- Diferenças entra o uso de bibliotecas dinâmicas e estáticas...

```
miguel@pegasus-linux:~$ gcc -Wall -static -L. gcc-ex2.c -o gcc-ex2 -lhello
miguel@pegasus-linux:~$ ll -h gcc-ex2
-rwxr-xr-x 1 miguel miguel 553K Mar 24 10:38 gcc-ex2
miguel@pegasus-linux:~$
miguel@pegasus-linux:~$ gcc -Wall -L. gcc-ex2.c -o gcc-ex2 -lhello
miguel@pegasus-linux:~$ ll -h gcc-ex2
-rwxr-xr-x 1 miguel miguel 6,6K Mar 24 10:38 gcc-ex2
```

Tamanho do executável é bem maior ao utilizar bibliotecas estáticas!

Warnings

- Opções adicionais de warnings
  - GCC provê muitas outras opções de warning que não são incluídas na opção –Wall
    - Muitas delas produzem warnings que indicam possíveis pontos problemáticos no código fonte. Exs.:
      - -W: Alerta sobre funções que podem retornar sem valor e comparações entre signed e unsigned
      - -Wconversion: Alerta sobre conversão entre float e int
      - -Wshadow: Alerta sobre redeclaração de variável em um mesmo escopo

Warnings

```c
#include <stdio.h>

int compara (unsigned int x) {
        if (x < 0)
                return 1;
        else
                return 0;
}

int main () {
        printf ("Resultado eh %d\n", compara (2));
        return 0;
}
```

```
miguel@pegasus-linux:~/UFRJ/disciplinas/linguagens/projetos$ gcc -W gcc-ex5.c -o gcc-ex5
gcc-ex5.c: In function ‘compara’:
gcc-ex5.c:4: warning: comparison of unsigned expression < 0 is always false
miguel@pegasus-linux:~/UFRJ/disciplinas/linguagens/projetos$ gcc -Wall gcc-ex5.c -o gcc-ex5
miguel@pegasus-linux:~/UFRJ/disciplinas/linguagens/projetos$ ./gcc-ex5
Resultado eh 0
```

Pré-processador

- Expande macros em arquivos fonte antes deles serem compilados
  - Opção –D: Define se uma macro está sendo usada
    - Quando uma macro é definida, o pré-processador insere o código correspondente até comando #endif
    - Inserida na linha de comando durante a compilação
      - Formato –D*NOME* (*NOME* é o nome da macro)

```c
#include <stdio.h>

int main () {
#ifdef TESTE
        printf ("Modo de TESTE\n");
#endif
        printf ("Rodando...\n");

        return 0;
}
```

```
miguel@pegasus-linux:~$ gcc -Wall gcc-ex6.c -o gcc-ex6
miguel@pegasus-linux:~$ ./gcc-ex6
Rodando...
miguel@pegasus-linux:~$ gcc -Wall -DTESTE gcc-ex6.c -o gcc-ex6
miguel@pegasus-linux:~$ ./gcc-ex6
Modo de TESTE
Rodando...
```

Pré-processador

- Macros podem ser também definidas com o #define no arquivo fonte ou nos arquivos de cabeçalho

```c
#include <stdio.h>

#define TESTE

int main () {
#ifdef TESTE
        printf ("Modo de TESTE\n");
#endif
        printf ("Rodando...\n");

        return 0;
}
```

```
miguel@pegasus-linux:~$ gcc -Wall gcc-ex6.c -o gcc-ex6
miguel@pegasus-linux:~$ ./gcc-ex6
Modo de TESTE
Rodando...
```

Pré-processador

- Macros também podem ser utilizadas passando valor
  - Formato **-DVAR=VALOR**

```c
#include <stdio.h>

int main () {
        printf ("Valor do numero eh %d\n", NUM);

        return 0;
}
```

```
miguel@pegasus-linux:~$ gcc -Wall gcc-ex7.c -o gcc-ex7
gcc-ex7.c: In function ‘main’:
gcc-ex7.c:4: error: ‘NUM’ undeclared (first use in this function)
gcc-ex7.c:4: error: (Each undeclared identifier is reported only once
gcc-ex7.c:4: error: for each function it appears in.)
miguel@pegasus-linux:~$ gcc -Wall -DNUM=5 gcc-ex7.c -o gcc-ex7
miguel@pegasus-linux:~$ ./gcc-ex7
Valor do numero eh 5
```

Pré-processador

- Opção: -E
  - Exibe as declarações das funções incluídas dos arquivos de cabeçalho pelo pré-processador
    - Comando grep filtra a saída do gcc

```
miguel@pegasus-linux:~$ gcc -E gcc-ex7.c | grep printf
extern int _IO_vfprintf (_IO_FILE *__restrict, const char *__restrict,
extern int fprintf (FILE *__restrict __stream,
extern int printf (__const char *__restrict __format, ...);
extern int sprintf (char *__restrict __s,
extern int vfprintf (FILE *__restrict __s, __const char *__restrict __format,
extern int vprintf (__const char *__restrict __format, __gnuc_va_list __arg);
extern int vsprintf (char *__restrict __s, __const char *__restrict __format,
extern int snprintf (char *__restrict __s, size_t __maxlen,
     __attribute__ ((__nothrow__)) __attribute__ ((__format__ (__printf__, 3, 4)
));
extern int vsnprintf (char *__restrict __s, size_t __maxlen,
     __attribute__ ((__nothrow__)) __attribute__ ((__format__ (__printf__, 3, 0)
));
 printf ("Valor do numero eh %d\n", NUM);
```

Compilação em Modo de Depuração

- Arquivo executável
  - Não contém referências ao código fonte do programa original
    - Exs.: nomes de variáveis, número de linhas etc.
  - É simplesmente uma sequência de instruções em código de máquina criado pelo compilador

Como então seria possível fazer depuração se não há como encontrar, de maneira simples, possíveis fontes de erro no código fonte?

Compilação em Modo de Depuração

- Opção: *-g*
  - Opção para depuração
    - Armazena informação adicional de depuração em arquivos executáveis e em arquivos objetos
    - Depuradores conseguem rastrear erros baseados em informações contidas nos arquivos compilados

Compilação em Modo de Depuração

- Opção: *-g*
  - Opção para depuração
    - Informações de nomes e linhas de código são armazenadas em tabelas de símbolos contidas nos arquivos compilados
    - Quando programa pára de maneira anormal, o compilador gera um arquivo chamado `core` que combinado com informações do modo de depuração auxiliam na busca da causa do erro
      - Informações da linha que provocou o erro e do valor das variáveis

Compilação em Modo de Depuração

```
int foo (int *p);

int main () {

    int *p = 0;
    return foo (p);
}

int foo (int *p) {


    int y = *p;
    return y;
}
```

Qual é o erro no código ao lado?

Compilação em Modo de Depuração

```c
int foo (int *p);

int main () {
    // Ponteiro inicializado com 0
    int *p = 0;
    return foo (p);
}

int foo (int *p) {
    // Função tenta desreferenciar um
    // ponteiro nulo, o que gera erro
    int y = *p;
    return y;
}
```

```
miguel@pegasus-linux:~$ gcc -Wall -g coreDump.c -o coreDump
miguel@pegasus-linux:~$ ./coreDump
Falha de segmentação
miguel@pegasus-linux:~$ ulimit -c
0
miguel@pegasus-linux:~$ ulimit -c unlimited
miguel@pegasus-linux:~$ ulimit -c
unlimited
miguel@pegasus-linux:~$ ./coreDump
Falha de segmentação (core dumped)
```

Compilação em Modo de Depuração

- **`ulimit -c`**
  - Exibe o tamanho permitido para arquivos do tipo core
- **`ulimit -c unlimited`**
  - Permite tamanho ilimitado para arquivos do tipo core

Depurador

- Programa gdb (GNU Debugger)
  - **gdb <arquivo_executável> <arquivo_core>**

```
miguel@pegasus-linux:~$ gdb coreDump core
GNU gdb 6.8-debian
Copyright (C) 2008 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "i486-linux-gnu"...

warning: Can't read pathname for load map: Input/output error.
Reading symbols from /lib/i686/cmov/libc.so.6...done.
Loaded symbols for /lib/i686/cmov/libc.so.6
Reading symbols from /lib/ld-linux.so.2...done.
Loaded symbols for /lib/ld-linux.so.2
Core was generated by `./coreDump'.
Program terminated with signal 11, Segmentation fault.
[New process 3204]
#0  0x080483a9 in foo (p=0x0) at coreDump.c:12
12              int y = *p;
(gdb) print y
$1 = 134518124
(gdb) print p
$2 = (int *) 0x0
```

Comandos gdb

- **`print <variável>`**
  - Imprime o valor da variável no momento da parada do programa
- **`trace`**
  - Imprime as chamadas de funções e os valores das variáveis até a parada do programa
    - Procedimento chamado de backtrace

Comandos gdb

```
This GDB was configured as "i486-linux-gnu"...

warning: Can't read pathname for load map: Input/output error.
Reading symbols from /lib/i686/cmov/libc.so.6...done.
Loaded symbols for /lib/i686/cmov/libc.so.6
Reading symbols from /lib/ld-linux.so.2...done.
Loaded symbols for /lib/ld-linux.so.2
Core was generated by `./coreDump'.
Program terminated with signal 11, Segmentation fault.
[New process 3204]
#0  0x080483a9 in foo (p=0x0) at coreDump.c:12
12              int y = *p;
(gdb) print y
$1 = 134518124
(gdb) print p
$2 = (int *) 0x0
(gdb) break main
Breakpoint 1 at 0x8048385: file coreDump.c, line 5.
(gdb) run
Starting program: /home/miguel/coreDump

Breakpoint 1, main () at coreDump.c:5
5               int *p = 0;
```

Comandos gdb

- Inserção de breakpoint
  - Execução do programa pára e retorna o controle para o depurador
    - **`break <função, linha ou posição de memória>`**
- Execução do programa
  - Programa começa a execução até encontrar o breakpoint
    - **`run`**
    - Para continuar a execução usar o **`step`** ou **`next`**
      - Comando step mostra a execução de qualquer função chamada na linha (maior nível de detalhes)

Comandos gdb

```
(gdb) print p
$2 = (int *) 0x0
(gdb) break main
Breakpoint 1 at 0x8048385: file coreDump.c, line 5.
(gdb) run
Starting program: /home/miguel/coreDump

Breakpoint 1, main () at coreDump.c:5
5               int *p = 0;
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/miguel/coreDump

Breakpoint 1, main () at coreDump.c:5
5               int *p = 0;
(gdb) next
6               return foo (p);
(gdb) next

Program received signal SIGSEGV, Segmentation fault.
0x080483a9 in foo (p=0x0) at coreDump.c:12
12              int y = *p;
```

Comandos gdb

- Valores de variáveis podem ser corrigidas
  - `set variable <variável> = <valor>`
    - Após isso o programa executará normalmente mesmo após utilizar o comando `step`

- **`finish`**
  - Continua a execução da função até o final, exibindo o valor de retorno encontrado

- **`continue`**
  - Continua a execução do programa até o final ou até o próximo breakpoint

Comandos gdb

- Programas com loop infinito podem ser também depurados
  - Primeiro executa o programa
  - Depois inicia o **gdb** para o programa em execução
    - **`attach <pid>`**: Anexa um determinado programa em execução ao gdb
      - Comando "**`ps x`**" obtém identificador do processo
    - **`kill`**: Mata o programa em execução

Comandos gdb

```c
int main () {
        unsigned int i = 0;

        while (1) i++;

        return 0;
}
```

```
miguel@pegasus-linux:~$ gcc -Wall -g gcc-ex9.c -o gcc-ex9
miguel@pegasus-linux:~$ ./gcc-ex9
Morto
```

```
miguel@pegasus-linux:/mnt/ufrj/disciplinas/linguagens/projetos$ gdb
GNU gdb 6.8-debian
Copyright (C) 2008 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "i486-linux-gnu"...
(gdb) attach 3850
Attaching to program: /mnt/ufrj/disciplinas/linguagens/projetos/gcc-ex9, process
 3850
Reading symbols from /lib/i686/cmov/libc.so.6...done.
Loaded symbols for /lib/i686/cmov/libc.so.6
Reading symbols from /lib/ld-linux.so.2...done.
Loaded symbols for /lib/ld-linux.so.2
0x08048390 in main () at gcc-ex9.c:4
4               while (1) i++;
(gdb) print i
$1 = 1349516083
(gdb) kill
Kill the program being debugged? (y or n) y
(gdb)
```

Comandos gdb

```c
int main () {
        unsigned int i = 0;

        while (1) i++;

        return 0;
}
```

```
miguel@pegasus-linux:~$ gcc -Wall -g gcc-ex9.c -o gcc-ex9
miguel@pegasus-linux:~$ ./gcc-ex9
Morto
```

```
miguel@pegasus-linux:/mnt/ufrj/disciplinas/linguagens/projetos$ gdb
GNU gdb 6.8-debian
Copyright (C) 2008 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "i486-linux-gnu"...
(gdb) attach 3850
Attaching to program: /mnt/ufrj/disciplinas/linguagens/projetos/gcc-ex9, process
 3850
Reading symbols from /lib/i686/cmov/libc.so.6...done.
Loaded symbols for /lib/i686/cmov/libc.so.6
Reading symbols from /lib/ld-linux.so.2...done.
Loaded symbols for /lib/ld-linux.so.2
0x08048390 in main () at gcc-ex9.c:4
4               while (1) i++;
(gdb) print i
$1 = 1349516083
(gdb) kill
Kill the program being debugged? (y or n) y
(gdb)
```

pid do processo

Otimizações

- O GCC é um compilador para fazer otimizações
  - Possui opções para aumentar a velocidade e reduzir o tamanho dos executáveis gerados
- Processo de otimização é complexo
  - Código em alto nível pode ser traduzido em diferentes combinações de instruções para atingir o mesmo resultado
  - Código gerado depende do processador
    - Ex.: há processadores que oferecem um número maior de registradores que outros

Otimização no Código

- Não requer conhecimento da arquitetura do processador
  - Dois tipos comuns:
    - Eliminação de expressão comum (*common subexpression elimination*)
    - Inserção de funções (*function inlining*)

Eliminação de Expressão Comum

- Consiste em calcular uma expressão no código fonte com menos instruções
  - Reusa resultados já calculados

```
x = cos(v)*(1 + sen(u/2)) + sen(w)*(1 - sen(u/2))

Pode ser reescrito, da seguinte maneira...

t = sen(u/2)
x = cos(v)*(1 + t) + sen(w)*(1 - t)
```

Inserção de Funções

- Sempre que uma função é chamada...
  - CPU armazena os argumentos da função em registradores e/ou posições de memória apropriados
  - Há um pulo para o início da função trazendo as páginas na memória virtual para a memória física ou cache
  - Início da execução do código da função
  - Retorno ao ponto original de execução após o término da função

Todo esse procedimento é chamado de sobrecarga de chamada de função que consome tempo de processamento!

Inserção de Funções

- Elimina a sobrecarga de chamada de função
  - Substitui a chamada pelo próprio código da função
  - É mais significativa caso a função chamada seja pequena
    - Código inserido no programa possui um número menor de instruções que o necessário para realizar a chamada

Exemplo de código cuja sobrecarga da chamada seria comparável ao tempo de execução necessário para realizar uma única multiplicação

```
double sq (double x) {
      return x*x;
}
```

Inserção de Funções

- Elimina a sobrecarga de chamada de função
  - Substitui a chamada pelo próprio código da função
  - É mais significativa caso a função chamada seja pequena
    - Código inserido no programa possui um número menor de instruções que o necessário para realizar a chamada

E se a função fosse usada dessa maneira???

```
for (i = 0; i < 10000000; i++) {
      sum += sq (i + 0.5);
}
```

Inserção de Funções

- Elimina a sobrecarga de chamada de função
  - Substitui a chamada pelo próprio código da função
  - É mais significativa caso a função chamada seja pequena
    - Código inserido no programa possui um número menor de instruções que o necessário para realizar a chamada

O procedimento de inserção (*inlining*) da função resulta no seguinte código otimizado:

```
for (i = 0; i < 10000000; i++) {
      double t = (i + 0.5); //Variável temporária
      sum += t*t;
}
```

Inserção de Funções

- O GCC seleciona funções para serem inseridas
  - Baseado no tamanho das funções
- Funções também podem ser solicitadas para serem inseridas explicitamente pelo programador
  - Uso da palavra-chave `inline`
    - Entretanto, compilador verifica se há a possibilidade

Compromisso entre Velocidade e Tamanho

- Algumas opções de otimização podem tornar o código:
  - Mais rápido, mas...
  - Maior em tamanho

Equivalentemente

  - Menor em tamanho, mas...
  - Mais lento

Ex.: desenrolamento de laços (*loop unrolling*)

Desenrolamento de Laços

- Aumenta a velocidade de execução do programa
  - Elimina a verificação da condição de término do laço em cada iteração

```
for (i = 0; i < 8; i++) {
      y[i] = i;
}
```

Otimização

```
y[0] = 0;
y[1] = 1;
y[2] = 2;
y[3] = 3;
y[4] = 4;
y[5] = 5;
y[6] = 6;
y[7] = 7;
```

Processo de otimização elimina a necessidade de checagem da condição de contorno e ainda permite paralelização das sentenças de atribuição. Entretanto, aumenta o tamanho do arquivo se o número de atribuições for maior que o tamanho do laço.

Agendamento

- Nível mais baixo de otimização
  - Compilador determina a melhor ordem das instruções individuais
    - Ordem deve ser tal que todas as instruções possuam os dados necessários antes da execução
- Não aumenta o tamanho do executável
  - Mas demora mais tempo para compilar devido a maior complexidade do procedimento

Níveis de Otimização

- Oferecidos pelo GCC para lidar com:
  - Tempo de compilação
  - Uso de memória
  - Compromisso entre velocidade e tamanho do executável

Níveis de Otimização

- Escolhidos com uma opção de linha de comando
  - Formato: `-O`***`NÍVEL`*** (Nível pode variar de 0 até 3)
    - `-O0`: *GCC* não realiza otimizações
    - `-O1`: *GCC* realiza as formas mais comuns de otimizações que não requerem compromisso entre velocidade e tamanho
    - `-O2`: *GCC* adiciona otimizações em relação ao 01 que incluem agendamento de instruções
    - `-O3`: *GCC* adiciona otimizações em relação ao 02 que incluem inserção de funções e procedimentos que aumentam a velocidade mas aumentam o tamanho do código. **Por esse motivo, normalmente o O2 se torna a melhor opção**

Níveis de Otimização

- Escolhidos com uma opção de linha de comando
  - Formato: `-O`***`NÍVEL`*** (Nível pode variar de 0 até 3)
    - funroll-loops: Independente das anteriores, habilita o desenrolamento de laços
    - `-Os`: *GCC* seleciona apenas otimizações que reduzem o tamanho do executável para plataformas com restrições de recursos

Níveis de Otimização

```c
#include <stdio.h>

double powern (double d, unsigned n) {
        double x = 1.0;
        unsigned j;

        for (j = 1; j <= n; j++)
                x *= d;

        return x;
}

int main () {
        double sum = 0.0;
        unsigned i;

        for (i = 1; i <= 100000000; i++) {
                sum += powern (i, i % 5);
        }

        printf ("sum = %g\n", sum);

        return 0;
}
```

Níveis de Otimização

```
miguel@pegasus-linux:~$ gcc -Wall -O0 gcc-ex8.c -o gcc-ex8 -lm
miguel@pegasus-linux:~$ time ./gcc-ex8
sum = 4e+38

real    0m9.179s
user    0m8.977s
sys     0m0.040s
miguel@pegasus-linux:~$ gcc -Wall -O1 gcc-ex8.c -o gcc-ex8 -lm
miguel@pegasus-linux:~$ time ./gcc-ex8
sum = 4e+38

real    0m6.977s
user    0m6.744s
sys     0m0.060s
miguel@pegasus-linux:~$ gcc -Wall -O2 gcc-ex8.c -o gcc-ex8 -lm
miguel@pegasus-linux:~$ time ./gcc-ex8
sum = 4e+38

real    0m5.855s
user    0m5.688s
sys     0m0.044s
```

user: tempo de execução do processo em CPU

total: tempo total para a execução do programa incluindo tempo de espera por CPU

sys: tempo esperando chamadas de sistema

Níveis de Otimização

```
miguel@pegasus-linux:~$ gcc -Wall -O3 gcc-ex8.c -o gcc-ex8 -lm
miguel@pegasus-linux:~$ time ./gcc-ex8
sum = 4e+38

real    0m5.840s
user    0m5.664s
sys     0m0.040s
miguel@pegasus-linux:~$ gcc -Wall -O3 -funroll-loops gcc-ex8.c -o gcc-ex8 -lm
miguel@pegasus-linux:~$ time ./gcc-ex8
sum = 4e+38

real    0m4.851s
user    0m4.700s
sys     0m0.024s
```

Os diferentes níveis de otimização permitiram uma queda de tempo de execução de quase 50% (-O0: 8,977s para –O3 –funroll-loops: 4,700s)
Em compensação, o tamanho do arquivo executável passou de 6,5k para 6,8k

Compilação de Programas em C++

- GCC compila códigos fonte em C++ diretamente em código de máquina
  - Outros compiladores convertem o código primeiro para C para depois compilar
- Procedimento é o mesmo para compilação em C
  - Uso do comando g++ ao invés do gcc
    - É parte do *GNU Compiler Collection*, assim como o gcc
    - Inclui bibliotecas típicas de C++
      - Ex.: iostream ao invés de stdio.h para E/S de dados
    - Possui mensagens de warning específicas
      - -Wall avisa sobre métodos e funções virtuais

Ferramentas Relacionadas ao Compilador

- Medida de desempenho: **gprof** (*GNU profiler*)
  - Grava o número de chamadas de cada função e o tempo gasto em cada uma delas
    - Esforços para melhorar o desempenho de um programa podem ser concentrados em funções que consumam muito tempo de processamento

```
gcc -Wall -pg <arquivo.c> -o <arquivo.o>

Opção -pg cria um arquivo com perfil "gmon.out"

gprof <arquivo_executável>
```

Ferramentas Relacionadas ao Compilador

```
miguel@pegasus-linux:~$ gcc -Wall -pg gcc-ex8.c -o gcc-ex8
miguel@pegasus-linux:~$ ./gcc-ex8
sum = 4e+38
miguel@pegasus-linux:~$ gprof gcc-ex8
Flat profile:

Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  ns/call  ns/call  name
 50.68      3.71     3.71 100000000    37.10    37.10  powern
 49.32      7.32     3.61                             main
```

Colunas:

1: Fração do tempo total gasto na função

2: Tempo cumulativo gasto na execução das funções calculado somando o tempo das anteriores

3: Tempo gasto na execução da função em particular

4: Número de chamadas

5: Tempo médio gasto por chamada em ns

6: Tempo gasto por chamada, contando o tempo de chamada às funções filhas em ns

```c
#include <stdio.h>

double powern (double d, unsigned n) {
        double x = 1.0;
        unsigned j;

        for (j = 1; j <= n; j++)
                x *= d;

        return x;
}

int main () {
        double sum = 0.0;
        unsigned i;

        for (i = 1; i <= 100000000; i++) {
                sum += powern (i, i % 5);
        }

        printf ("sum = %g\n", sum);

        return 0;
}
```

Ferramentas Relacionadas ao Compilador

- Medida de desempenho: **gcov** (*GNU coverage*)
  - Mostra o número de vezes que cada linha do programa é executada
    - Permite encontrar áreas de código que não são usadas
    - Combinada com o **gprof** permite concentrar ainda mais os esforços para melhorar o desempenho do programa

```
gcc -Wall -fprofile-arcs -ftest-coverage <arquivo.c>
```

Opção -ftest-coverage adiciona instruções para contar o número de vezes que cada linha é executada e o -fprofile-arcs adiciona código para contar o número de vezes que a execução do programa entra em cada ramificação do código. Gera arquivos com informações usados pelo gcov.

Executar e depois rodar `gcov <arquivo.c>`

Cria o arquivo arquivo.c.gcov que contém o número de vezes que cada linha é executada.

Ferramentas Relacionadas ao Compilador

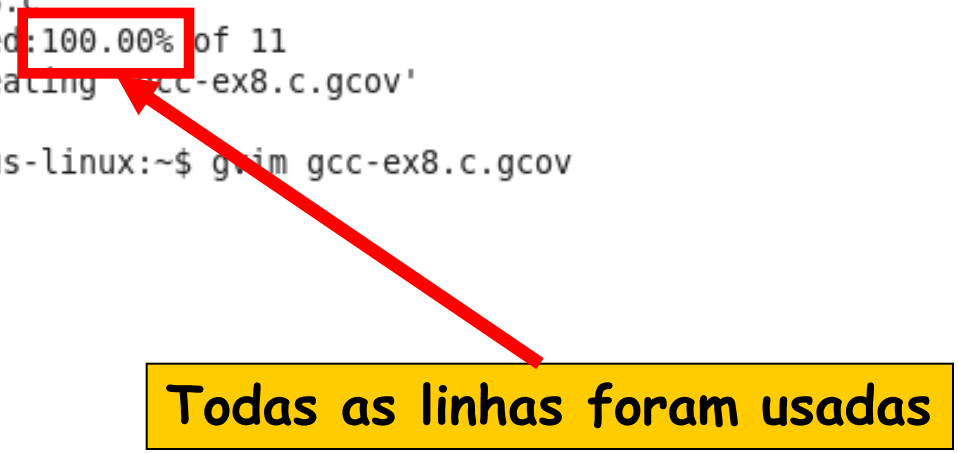
```
miguel@pegasus-linux:~$ gcc -Wall -fprofile-arcs -ftest-coverage gcc-ex8.c -o gcc-ex8
miguel@pegasus-linux:~$ ./gcc-ex8
sum = 4e+38
miguel@pegasus-linux:~$ gcov gcc-ex8.c
File 'gcc-ex8.c'
Lines executed:100.00% of 11
gcc-ex8.c:creating 'gcc-ex8.c.gcov'

miguel@pegasus-linux:~$ gvim gcc-ex8.c.gcov
```

Ferramentas Relacionadas ao Compilador

```
miguel@pegasus-linux:~$ gcc -Wall -fprofile-arcs -ftest-coverage gcc-ex8.c -o gcc-ex8
miguel@pegasus-linux:~$ ./gcc-ex8
sum = 4e+38
miguel@pegasus-linux:~$ gcov gcc-ex8.c
File 'gcc-ex8.c'
Lines executed:100.00% of 11
gcc-ex8.c:creating 'gcc-ex8.c.gcov'

miguel@pegasus-linux:~$ gvim gcc-ex8.c.gcov
```



Ferramentas Relacionadas ao Compilador

Arquivo: gcc-ex8.c.cov

```
        -:    0:Source:gcc-ex8.c
        -:    0:Graph:gcc-ex8.gcno
        -:    0:Data:gcc-ex8.gcda
        -:    0:Runs:1
        -:    0:Programs:1
        -:    1:#include <stdio.h>
        -:    2:
100000000:    3:double powern (double d, unsigned n) {
100000000:    4:        double x = 1.0;
        -:    5:        unsigned j;
        -:    6:
300000000:    7:        for (j = 1; j <= n; j++)
200000000:    8:                x *= d;
        -:    9:
100000000:   10:        return x;
        -:   11:}
        -:   12:
        1:   13:int main () {
        1:   14:        double sum = 0.0;
        -:   15:        unsigned i;
        -:   16:
100000001:   17:        for (i = 1; i <= 100000000; i++) {
100000000:   18:                sum += powern (i, i % 5);
        -:   19:        }
        -:   20:
        1:   21:        printf ("sum = %g\n", sum);
        -:   22:
        1:   23:        return 0;
        -:   24:}
```

Ferramentas Relacionadas ao Compilador

```c
#include <stdio.h>

int main () {
        int i;
        for (i = 1; i < 10; i++) {
                if (i % 3 == 0)
                        printf ("%d eh divisivel por 3\n", i);
                if (i % 11 == 0)
                        printf ("%d eh divisivel por 11\n", i);
        }
        return 0;
}
```

```
miguel@pegasus-linux:~$ gcc -Wall -fprofile-arcs -ftest-coverage gcc-ex10.c -o gcc-ex10
miguel@pegasus-linux:~$ ./gcc-ex10
3 eh divisivel por 3
6 eh divisivel por 3
9 eh divisivel por 3
miguel@pegasus-linux:~$ gcov gcc-ex10.c
File 'gcc-ex10.c'
Lines executed:85.71% of 7
gcc-ex10.c:creating 'gcc-ex10.c.gcov'

miguel@pegasus-linux:~$ gvim gcc-ex10.c.gcov
```

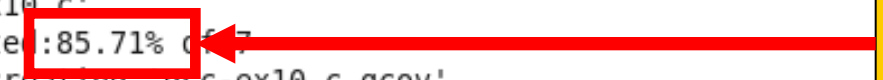
Ferramentas Relacionadas ao Compilador

```c
#include <stdio.h>

int main () {
        int i;
        for (i = 1; i < 10; i++) {
                if (i % 3 == 0)
                        printf ("%d eh divisivel por 3\n", i);
                if (i % 11 == 0)
                        printf ("%d eh divisivel por 11\n", i);
        }
        return 0;
}
```

```
miguel@pegasus-linux:~$ gcc -Wall -fprofile-arcs -ftest-coverage gcc-ex10.c -o gcc-ex10
miguel@pegasus-linux:~$ ./gcc-ex10
3 eh divisivel por 3
6 eh divisivel por 3
9 eh divisivel por 3
miguel@pegasus-linux:~$ gcov gcc-ex10.c
File 'gcc-ex10.c'
Lines executed:85.71% of 7
gcc-ex10.c:creating 'gcc-ex10.c.gcov'

miguel@pegasus-linux:~$ gvim gcc-ex10.c.gcov
```



Nem todas as linhas foram usadas

Ferramentas Relacionadas ao Compilador

```
    -:    0:Source:gcc-ex10.c
    -:    0:Graph:gcc-ex10.gcno
    -:    0:Data:gcc-ex10.gcda
    -:    0:Runs:1
    -:    0:Programs:1
    -:    1:#include <stdio.h>
    -:    2:
    1:    3:int main () {
    -:    4:        int i;
   10:    5:        for (i = 1; i < 10; i++) {
    9:    6:                if (i % 3 == 0)
    3:    7:                        printf ("%d eh divisivel por 3\n", i);
    9:    8:                if (i % 11 == 0)
#####:    9:                        printf ("%d eh divisivel por 11\n", i);
    -:   10:        }
    1:   11:        return 0;
    -:   12:}
    -:   13:
```

Ferramentas Relacionadas ao Compilador

```
    -:    0:Source:gcc-ex10.c
    -:    0:Graph:gcc-ex10.gcno
    -:    0:Data:gcc-ex10.gcda
    -:    0:Runs:1
    -:    0:Programs:1
    -:    1:#include <stdio.h>
    -:    2:
    1:    3:int main () {
    -:    4:        int i;
   10:    5:        for (i = 1; i < 10; i++) {
    9:    6:                if (i % 3 == 0)
    3:    7:                        printf ("%d eh divisivel por 3\n", i);
    -:    8:                if (i % 11 == 0)
#####:    9:                        printf ("%d eh divisivel por 11\n", i);
    -:   10:        }
    1:   11:        return 0;
    -:   12:}
    -:   13:
```

Linhas não usada

Ferramentas Relacionadas ao Compilador

Transformando para ficar mais parecido com C++...

```
#include <iostream>

using namespace std;

int main () {
        int i;
        for (i = 1; i < 10; i++) {
                if (i % 3 == 0)
                        cout << i << " eh divisivel por 3\n";
                if (i % 11 == 0)
                        cout << i << " eh divisivel por 11\n";
        }
        return 0;
}
```

Ferramentas Relacionadas ao Compilador

```
miguel@pegasus-linux:~$ g++ -Wall -fprofile-arcs -ftest-coverage gcc-ex10.cpp
miguel@pegasus-linux:~$ ./gcc-ex10
3 eh divisivel por 3
6 eh divisivel por 3
9 eh divisivel por 3
miguel@pegasus-linux:~$ gcov gcc-ex10.cpp
File '/usr/include/c++/4.3/bits/ios_base.h'
Lines executed:0.00% of 2
/usr/include/c++/4.3/bits/ios_base.h:creating 'ios_base.h.gcov'

File '/usr/include/c++/4.3/bits/basic_ios.h'
Lines executed:0.00% of 4
/usr/include/c++/4.3/bits/basic_ios.h:creating 'basic_ios.h.gcov'

File 'gcc-ex10.cpp'
Lines executed:87.50% of 8
gcc-ex10.cpp:creating 'gcc-ex10.cpp.gcov'

File '/usr/include/c++/4.3/iostream'
Lines executed:100.00% of 1
/usr/include/c++/4.3/iostream:creating 'iostream.gcov'

File '/usr/include/c++/4.3/bits/char_traits.h'
Lines executed:0.00% of 2
/usr/include/c++/4.3/bits/char_traits.h:creating 'char_traits.h.gcov'

File '/usr/include/c++/4.3/ostream'
Lines executed:0.00% of 5
/usr/include/c++/4.3/ostream:creating 'ostream.gcov'

miguel@pegasus-linux:~$ gvim gcc-ex10.cpp.gcov
```

Ferramentas Relacionadas ao Compilador

```
    -:    0:Source:gcc-ex10.cpp
    -:    0:Graph:gcc-ex10.gcno
    -:    0:Data:gcc-ex10.gcda
    -:    0:Runs:1
    -:    0:Programs:1
    -:    1:#include <iostream>
    -:    2:
    -:    3:using namespace std;
    -:    4:
    1:    5:int main () {
    -:    6:        int i;
   10:    7:        for (i = 1; i < 10; i++) {
    9:    8:                if (i % 3 == 0)
    3:    9:                        cout << i << " eh divisivel por 3\n";
    9:   10:                if (i % 11 == 0)
#####:   11:                        cout << i << " eh divisivel por 11\n";
    -:   12:        }
    1:   13:        return 0;
    3:   14:}
    -:   15:
```

Ferramentas Relacionadas ao Compilador

- Medida de desempenho: **valgrind**
  - Conjunto de programas para depurar e avaliar o desempenho do programa
    - A ferramenta mais utilizada é a **Memcheck** para avaliar erros comuns de memória

```
gcc -Wall -g <arquivo.c> -o <arquivo.o>

Programa deve ser compilado em modo de depuração

valgrind --leak-check=yes <arquivo_executável>

A ferramenta Memcheck é usada por padrão. A opção leak-check liga o detector de vazamento de memória
```

Ferramentas Relacionadas ao Compilador

```c
#include <stdlib.h>

void f(void) {
        int *x = malloc(10 * sizeof(int));
        x[10] = 0;
}

int main(void) {
        f();
        return 0;
}
```

Quais são os problemas desse programa?

Ferramentas Relacionadas ao Compilador

```c
#include <stdlib.h>

void f(void) {
        int *x = malloc(10 * sizeof(int));
        x[10] = 0;      // problema 1: Alocação de memória errada
}                       // problema 2: Vazamento de memória, x não foi liberado

int main(void) {
        f();
        return 0;
}
```

Quais são os problemas desse programa?

Ferramentas Relacionadas ao Compilador

```c
#include <stdlib.h>

void f(void) {
        int *x = malloc(10 * sizeof(int));
        x[10] = 0;      // problema 1: Alocação de memória errada
}                       // problema 2: Vazamento de memória, x não foi liberado

int main(void) {
        f();
        return 0;
}
```

```
miguel@pegasus-linux:~$ gcc -Wall -g gcc-ex13.c -o gcc-ex13
miguel@pegasus-linux:~$ valgrind --leak-check=yes ./gcc-ex13
==4450== Memcheck, a memory error detector.
==4450== Copyright (C) 2002-2007, and GNU GPL'd, by Julian Seward et al.
==4450== Using LibVEX rev 1854, a library for dynamic binary translation.
==4450== Copyright (C) 2004-2007, and GNU GPL'd, by OpenWorks LLP.
==4450== Using valgrind-3.3.1-Debian, a dynamic binary instrumentation framework.
==4450== Copyright (C) 2000-2007, and GNU GPL'd, by Julian Seward et al.
==4450== For more details, rerun with: -v
==4450==
==4450== Invalid write of size 4
==4450==    at 0x80483BF: f (gcc-ex13.c:5)
==4450==    by 0x80483DC: main (gcc-ex13.c:9)
==4450==  Address 0x4194050 is 0 bytes after a block of size 40 alloc'd
==4450==    at 0x4023D6E: malloc (vg_replace_malloc.c:207)
==4450==    by 0x80483B5: f (gcc-ex13.c:4)
==4450==    by 0x80483DC: main (gcc-ex13.c:9)
==4450==
==4450== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 13 from 1)
==4450== malloc/free: in use at exit: 40 bytes in 1 blocks.
==4450== malloc/free: 1 allocs, 0 frees, 40 bytes allocated.
==4450== For counts of detected errors, rerun with: -v
==4450== searching for pointers to 1 not-freed blocks.
==4450== checked 60,204 bytes.
```

Ferramentas Relacionadas ao Compilador

```c
#include <stdlib.h>

void f(void) {
        int *x = malloc(10 * sizeof(int));
        x[10] = 0;      // problema 1: Alocação de memória errada
}                       // problema 2: Vazamento de memória, x não foi liberado

int main(void) {
        f();
        return 0;
}
```

Erro de alocação detectado!

```
miguel@pegasus-linux:~$ gcc -Wall -g gcc-ex13.c -o gcc-ex13
miguel@pegasus-linux:~$ valgrind --leak-check=yes ./gcc-ex13
==4450== Memcheck, a memory error detector.
==4450== Copyright (C) 2002-2007, and GNU GPL'd, by Julian Seward et al.
==4450== Using LibVEX rev 1854, a library for dynamic binary translation.
==4450== Copyright (C) 2004-2007, and GNU GPL'd, by OpenWorks LLP.
==4450== Using valgrind-3.3.1-Debian, a dynamic binary instrumentation framework.
==4450== Copyright (C) 2000-2007, and GNU GPL'd, by Julian Seward et al.
==4450== For more details, rerun with: -v
==4450==
==4450== Invalid write of size 4
==4450==    at 0x80483BF: f (gcc-ex13.c:5)
==4450==    by 0x80483DC: main (gcc-ex13.c:9)
==4450==  Address 0x4194050 is 0 bytes after a block of size 40 alloc'd
==4450==    at 0x4023D6E: malloc (vg_replace_malloc.c:207)
==4450==    by 0x80483B5: f (gcc-ex13.c:4)
==4450==    by 0x80483DC: main (gcc-ex13.c:9)
==4450==
==4450== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 13 from 1)
==4450== malloc/free: in use at exit: 40 bytes in 1 blocks.
==4450== malloc/free: 1 allocs, 0 frees, 40 bytes allocated.
==4450== For counts of detected errors, rerun with: -v
==4450== searching for pointers to 1 not-freed blocks.
==4450== checked 60,204 bytes.
```

Ferramentas Relacionadas ao Compilador

```c
#include <stdlib.h>

void f(void) {
        int *x = malloc(10 * sizeof(int));
        x[10] = 0;      // problema 1: Alocação de memória errada
}                       // problema 2: Vazamento de memória, x não foi liberado

int main(void) {
        f();
        return 0;
}
```

```
==4450==
==4450==
==4450== 40 bytes in 1 blocks are definitely lost in loss record 1 of 1
==4450==    at 0x4023D6E: malloc (vg_replace_malloc.c:207)
==4450==    by 0x80483B5: f (gcc-ex13.c:4)
==4450==    by 0x80483DC: main (gcc-ex13.c:9)
==4450==
==4450== LEAK SUMMARY:
==4450==    definitely lost: 40 bytes in 1 blocks.
==4450==      possibly lost: 0 bytes in 0 blocks.
==4450==    still reachable: 0 bytes in 0 blocks.
==4450==         suppressed: 0 bytes in 0 blocks.
```

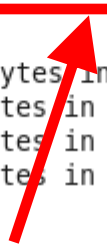
Ferramentas Relacionadas ao Compilador

```c
#include <stdlib.h>

void f(void) {
        int *x = malloc(10 * sizeof(int));
        x[10] = 0;      // problema 1: Alocação de memória errada
}                       // problema 2: Vazamento de memória, x não foi liberado

int main(void) {
        f();
        return 0;
}
```

```
==4450==
==4450==
==4450== 40 bytes in 1 blocks are definitely lost in loss record 1 of 1
==4450==    at 0x4023D6E: malloc (vg_replace_malloc.c:207)
==4450==    by 0x80483B5: f (gcc-ex13.c:4)
==4450==    by 0x80483DC: main (gcc-ex13.c:9)
==4450==
==4450== LEAK SUMMARY:
==4450==    definitely lost: 40 bytes in 1 blocks.
==4450==      possibly lost: 0 bytes in 0 blocks.
==4450==    still reachable: 0 bytes in 0 blocks.
==4450==         suppressed: 0 bytes in 0 blocks.
```



Erro de vazamento detectado!
O "definitely lost" significa que o programa está realmente vazando memória, se ele tivesse dúvida, colocaria "probably lost"

Processo de Compilação

- Processo com múltiplos estágios
  - Envolve diferentes ferramentas
    - Próprio compilador (`gcc` ou `g++`)
    - Montador (`as`)
    - Ligador (`ld`)
  - O processo completo ao se invocar o compilador é:
    - Pré-processamento ➔ compilação ➔ montagem ➔ ligação
      - Processo completo pode ser visto com a opção `-v`

Processo de Compilação

Pré-processamento:
`cpp <arquivo.c> > <arquivo.i>`

```c
#include <stdio.h>

int main () {
        printf ("Hello World!\n");
        return 0;
}
```

```c
extern int feof (FILE *__stream) __attribute__ ((__nothrow__)) ;
extern int ferror (FILE *__stream) __attribute__ ((__nothrow__)) ;
extern void clearerr_unlocked (FILE *__stream) __attribute__ ((__nothrow__));
extern int feof_unlocked (FILE *__stream) __attribute__ ((__nothrow__)) ;
extern int ferror_unlocked (FILE *__stream) __attribute__ ((__nothrow__)) ;
extern void perror (__const char *__s);
# 1 "/usr/include/bits/sys_errlist.h" 1 3 4
# 27 "/usr/include/bits/sys_errlist.h" 3 4
extern int sys_nerr;
extern __const char *__const sys_errlist[];
# 823 "/usr/include/stdio.h" 2 3 4
extern int fileno (FILE *__stream) __attribute__ ((__nothrow__)) ;
extern int fileno_unlocked (FILE *__stream) __attribute__ ((__nothrow__)) ;
# 842 "/usr/include/stdio.h" 3 4
extern FILE *popen (__const char *__command, __const char *__modes) ;
extern int pclose (FILE *__stream);
extern char *ctermid (char *__s) __attribute__ ((__nothrow__));
# 882 "/usr/include/stdio.h" 3 4
extern void flockfile (FILE *__stream) __attribute__ ((__nothrow__));
extern int ftrylockfile (FILE *__stream) __attribute__ ((__nothrow__)) ;
extern void funlockfile (FILE *__stream) __attribute__ ((__nothrow__));
# 912 "/usr/include/stdio.h" 3 4

# 2 "gcc-ex11.c" 2

int main () {
 printf ("Hello World!\n");
 return 0;
}
```

Processo de Compilação

Compilação:
`gcc -Wall -S <arquivo.i>`

```
extern int feof (FILE *__stream) __attribute__ ((__nothrow__)) ;
extern int ferror (FILE *__stream) __attribute__ ((__nothrow__)) ;
extern void clearerr_unlocked (FILE *__stream) __attribute__ ((__nothrow__));
extern int feof_unlocked (FILE *__stream) __attribute__ ((__nothrow__)) ;
extern int ferror_unlocked (FILE *__stream) __attribute__ ((__nothrow__)) ;
extern void perror (__const char *__s);
# 1 "/usr/include/bits/sys_errlist.h" 1 3 4
# 27 "/usr/include/bits/sys_errlist.h" 3 4
extern int sys_nerr;
extern __const char *__const sys_errlist[];
# 823 "/usr/include/stdio.h" 2 3 4
extern int fileno (FILE *__stream) __attribute__ ((__nothrow__)) ;
extern int fileno_unlocked (FILE *__stream) __attribute__ ((__nothrow__)) ;
# 842 "/usr/include/stdio.h" 3 4
extern FILE *popen (__const char *__command, __const char *__modes) ;
extern int pclose (FILE *__stream);
extern char *ctermid (char *__s) __attribute__ ((__nothrow__));
# 882 "/usr/include/stdio.h" 3 4
extern void flockfile (FILE *__stream) __attribute__ ((__nothrow__));
extern int ftrylockfile (FILE *__stream) __attribute__ ((__nothrow__)) ;
extern void funlockfile (FILE *__stream) __attribute__ ((__nothrow__));
# 912 "/usr/include/stdio.h" 3 4

# 2 "gcc-ex11.c" 2

int main () {
 printf ("Hello World!\n");
 return 0;
}
```

```
        .file   "gcc-ex11.c"
        .section        .rodata
.LC0:
        .string "Hello World!"
        .text
.globl main
        .type   main, @function
main:
        leal    4(%esp), %ecx
        andl    $-16, %esp
        pushl   -4(%ecx)
        pushl   %ebp
        movl    %esp, %ebp
        pushl   %ecx
        subl    $4, %esp
        movl    $.LC0, (%esp)
        call    puts
        movl    $0, %eax
        addl    $4, %esp
        popl    %ecx
        popl    %ebp
        leal    -4(%ecx), %esp
        ret
        .size   main, .-main
        .ident  "GCC: (Debian 4.3.2-1.1) 4.3.2"
        .section        .note.GNU-stack,"",@progbits
```

Processo de Compilação

```
        .file   "gcc-ex11.c"
        .section        .rodata
.LC0:
        .string "Hello World!"
        .text
.globl main
        .type   main, @function
main:
        leal    4(%esp), %ecx
        andl    $-16, %esp
        pushl   -4(%ecx)
        pushl   %ebp
        movl    %esp, %ebp
        pushl   %ecx
        subl    $4, %esp
        movl    $.LC0, (%esp)
        call    puts
        movl    $0, %eax
        addl    $4, %esp
        popl    %ecx
        popl    %ebp
        leal    -4(%ecx), %esp
        ret
        .size   main, .-main
        .ident  "GCC: (Debian 4.3.2-1.1) 4.3.2"
        .section        .note.GNU-stack,"",@progbits
```

Montagem:
`as <arquivo.s> -o <arquivo.o>`

Arquivo .o em linguagem de máquina

Processo de Compilação

Ligação:
`gcc <arquivo.o>`

Arquivo .o → **Executável**

```
miguel@pegasus-linux:~$ cpp gcc-ex11.c > gcc-ex11.i
miguel@pegasus-linux:~$ gcc -Wall -S gcc-ex11.i
miguel@pegasus-linux:~$ as gcc-ex11.s -o gcc-ex11.o
miguel@pegasus-linux:~$ gcc gcc-ex11.o -o gcc-ex11
miguel@pegasus-linux:~$ ./gcc-ex11
Hello World!
```

Automake

- Parte de um conjunto de ferramentas chamadas
  - *Autotools*
    - Geram arquivos como: configure, configure.ac, Makefile.in, Makefile.am, aclocal.m4 etc.
- Ferramenta para gerar automaticamente arquivos do tipo Makefile
  - Manutenção de Makefiles pode se tornar complexa
    - Makefile é reescrito sempre que o programa for compilado em uma plataformas diferente...

Processo ficaria mais simples se houvesse uma ferramenta que automaticamente ajustasse o Makefile!

Automake

- Processo completo de instalação, normalmente, compreende os seguintes passos:
  - Executar: `configure`
    - Verifica o sistema e gera o Makefile
  - Executar: `make`
    - Compila o código fonte do programa e mais das bibliotecas
  - Executar: `make check`
    - Roda testes para verificar os arquivos compilados
  - Executar: `make install`
    - Copia os cabeçalhos e bibliotecas para diretórios padrão
  - Executar: `make installcheck`
    - Roda testes para verificar a instalação

Arquivo de Configuração

- Verifica se o sistema possui instalações necessárias e ainda cria o arquivo Makefile
  - Passagem de parâmetro para o configure:
    - Sobrescreve a configuração padrão

```
Exs.: ./configure CC=gcc-3
      ./configure CFLAGS='-O2 -funroll-loops'
```

  - Compilação cruzada (*cross compilation*)
    - Compilar um programa em uma plataforma para ser usada em outra diferente

```
Ex.: ./configure --build i686-pc-linux-gnu --host i586-mingw32msvc
```

Opção **–build**: Sistema onde o pacote é construído
Opção **–host**: Sistema onde o pacote será executado

Criando um Pacote

- Passo 1: Criar um diretório para armazenar os fontes
  - Isso facilita a criação de outros diretórios para armazenar outros arquivos relacionados
    - Ex.: **/man** para manual, **data/** para dados etc.

```
mkdir gcc-ex12/src
```

    - Arquivo: **gcc-ex12/src/main.c**

```c
#include <config.h>
#include <stdio.h>

int main () {
        printf ("Hello World!\n");
        return 0;
}
```

Criando um Pacote

- Passo 2: Criar um arquivo do tipo **README**

```
This is a demonstration package for GNU Automake.
Type `info Automake' to read the Automake manual.
```

- Passo 3: Criar os arquivos **Makefile.am** e **src/Makefile.am**
  - Arquivos contêm instruções para o Automake
    - Contém lista de definições de variáveis

```
SUBDIRS = src
dist_doc_DATA = README
```

Makefile.am

```
bin_PROGRAMS = hello
hello_SOURCES = main.c
```

src/Makefile.am

Criando um Pacote

- Passo 4: Criar o arquivo **configure.ac**
  - Contém instruções para o Autoconf criar o arquivo de configuração (**configure**)

```
AC_INIT([amhello], [1.0], [bug-automake@gnu.org])
AM_INIT_AUTOMAKE([-Wall -Werror foreign])
AC_PROG_CC
AC_CONFIG_HEADERS([config.h])
AC_CONFIG_FILES([
 Makefile
 src/Makefile
])
AC_OUTPUT
```

Criando um Pacote

- Passo 4: Criar o arquivo **`configure.ac`**
  - **`AC_*`** **`e`** **`AM_*`** são macros do Autoconf e Automake, respectivamente
    - **`AC_INIT`** recebe o nome do pacote, versão e um endereço para reportar bugs
    - **`AM_INIT_AUTOMAKE`** são opções do Automake
    - **`AC_PROG_CC`** faz com que o compilador C seja buscado
    - **`AC_CONFIG_HEADERS`** cria um arquivo config.h contendo macros
    - **`AC_CONFIG_FILES`** recebe a lista dos arquivos que o configure deve criar
    - **`AC_OUTPUT`** é um comando de encerramento

Criando um Pacote

- Passo 5: Executar o comando **autoreconf**
  - Chama o **autoconf** para criar o configure e o **automake** para criar o Makefile

```
miguel@pegasus-linux:~/gcc-ex12$ autoreconf --install
configure.ac:2: installing `./install-sh'
configure.ac:2: installing `./missing'
src/Makefile.am: installing `./depcomp'
```

depcomp: Procura as dependência necessárias para a compilação

install-sh: Substitui o arquivo de instalação caso este não esteja disponível ou não possa ser usado

missing: Avisa problemas durante a instalação e tenta resolver, ex. pacotes faltantes

Criando um Pacote

- Após o Passo 5, o sistema está completo
  - **Começa a instalação!**
- Passo 6: Rodar o arquivo **`configure`**
  - Gerado no passo 5

```
miguel@pegasus-linux:~/gcc-ex12$ ./configure
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /bin/mkdir -p
checking for gawk... no
checking for mawk... mawk
checking whether make sets $(MAKE)... yes
checking for gcc... gcc
checking for C compiler default output file name... a.out
checking whether the C compiler works... yes
checking whether we are cross compiling... no
checking for suffix of executables...
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking for style of include used by make... GNU
checking dependency style of gcc... gcc3
configure: creating ./config.status
config.status: creating Makefile
config.status: creating src/Makefile
config.status: creating config.h
config.status: executing depfiles commands
miguel@pegasus-linux:~/gcc-ex12$ make
make  all-recursive
```

Criando um Pacote

- Passo 7: Rodar **`make.`** **Começa a compilação!**

```
miguel@pegasus-linux:~/gcc-ex12$ make
make  all-recursive
make[1]: Entrando no diretório `/home/miguel/gcc-ex12'
Making all in src
make[2]: Entrando no diretório `/home/miguel/gcc-ex12/src'
gcc -DHAVE_CONFIG_H -I. -I..     -g -O2 -MT main.o -MD -MP -MF .deps/main.Tpo -c -o main.o main.c
mv -f .deps/main.Tpo .deps/main.Po
gcc  -g -O2   -o hello main.o
make[2]: Saindo do diretório `/home/miguel/gcc-ex12/src'
make[2]: Entrando no diretório `/home/miguel/gcc-ex12'
make[2]: Nada a ser feito para `all-am'.
make[2]: Saindo do diretório `/home/miguel/gcc-ex12'
make[1]: Saindo do diretório `/home/miguel/gcc-ex12'
```

- Passo 8: Programa já pode ser executado!
  - **`./src/hello`**
- Passo 9: Rodar **`make –distcheck`**
  - Empacota o programa em um **`tar.gz`**

Criando um Pacote

```
miguel@pegasus-linux:~/gcc-ex12$ make distcheck
{ test ! -d amhello-1.0 || { find amhello-1.0 -type d ! -perm -200 -exec chmod u+w {} ';' && rm -fr amhello
-1.0; }; }
test -d amhello-1.0 || mkdir amhello-1.0
list='src'; for subdir in $list; do \
          if test "$subdir" = .; then :; else \
            test -d "amhello-1.0/$subdir" \
            || /bin/mkdir -p "amhello-1.0/$subdir" \
            || exit 1; \
            distdir=`CDPATH="${ZSH_VERSION+.}:" && cd amhello-1.0 && pwd`; \
            top_distdir=`CDPATH="${ZSH_VERSION+.}:" && cd amhello-1.0 && pwd`; \
            (cd $subdir && \
              make  \
                top_distdir="$top_distdir" \
                distdir="$distdir/$subdir" \
```

• • •

```
test -z "" || rm -f
rm -f config.h stamp-h1
rm -f TAGS ID GTAGS GRTAGS GSYMS GPATH tags
make[2]: Saindo do diretório `/home/miguel/gcc-ex12/amhello-1.0/_build'
rm -f config.status config.cache config.log configure.lineno config.status.lineno
rm -f Makefile
make[1]: Saindo do diretório `/home/miguel/gcc-ex12/amhello-1.0/_build'
{ test ! -d amhello-1.0 || { find amhello-1.0 -type d ! -perm -200 -exec chmod u+w {} ';' && rm -fr amhello
-1.0; }; }
============================================
amhello-1.0 archives ready for distribution:
amhello-1.0.tar.gz
============================================
```

Doxygen

- Gera documentação do próprio código-fonte em diferentes linguagens
  - Inclusive C, C++ e Python
    - Documentação online (*.html) e offline (usando Latex)
  - Requer configuração para extrair a documentação desejada

Exemplo de Programa de Captura de Pacotes

- A ideia é criar um programa para captura de pacotes
  - Requer uso da biblioteca **`libpcap`**
    - Mesma usada nos programas que fazem farejamento de redes ("*packet sniffers*")
      - Ex.: tcpdump, wireshark, etc.
    - Instalação via apt-get:
      - **`apt-get install libpcap-dev`**
    - Documentação disponível em:
      - **`http://www.tcpdump.org/pcap.html`**

Programa de Captura em Execução

```
root@itaqua:/lab/users/Miguel/miguel/disciplinas/progredes/libpcap# ./captura
Device: eth0
*** Press ctrl-c to finish ***
[2017-06-27 10:13:49.901723] Got packet of 86 bytes
[2017-06-27 10:13:49.901739] Got packet of 54 bytes
[2017-06-27 10:13:49.937981] Got packet of 278 bytes
[2017-06-27 10:13:49.938115] Got packet of 138 bytes
[2017-06-27 10:13:49.938127] Got packet of 66 bytes
[2017-06-27 10:13:49.938284] Got packet of 242 bytes
[2017-06-27 10:13:49.938385] Got packet of 138 bytes
[2017-06-27 10:13:49.938553] Got packet of 234 bytes
[2017-06-27 10:13:49.938775] Got packet of 138 bytes
[2017-06-27 10:13:49.938844] Got packet of 206 bytes
[2017-06-27 10:13:49.939000] Got packet of 258 bytes
[2017-06-27 10:13:49.939075] Got packet of 222 bytes
[2017-06-27 10:13:49.939279] Got packet of 8192 bytes
[2017-06-27 10:13:49.939299] Got packet of 66 bytes
[2017-06-27 10:13:49.939374] Got packet of 4410 bytes
[2017-06-27 10:13:49.939379] Got packet of 66 bytes
[2017-06-27 10:13:49.939613] Got packet of 2034 bytes
[2017-06-27 10:13:49.939620] Got packet of 66 bytes
[2017-06-27 10:13:49.939677] Got packet of 206 bytes
[2017-06-27 10:13:49.939892] Got packet of 258 bytes
[2017-06-27 10:13:49.939959] Got packet of 242 bytes
[2017-06-27 10:13:49.940095] Got packet of 138 bytes
[2017-06-27 10:13:49.940272] Got packet of 242 bytes
[2017-06-27 10:13:49.940432] Got packet of 138 bytes
[2017-06-27 10:13:49.940507] Got packet of 242 bytes
[2017-06-27 10:13:49.940624] Got packet of 138 bytes
[2017-06-27 10:13:49.940694] Got packet of 242 bytes
[2017-06-27 10:13:49.940832] Got packet of 138 bytes
[2017-06-27 10:13:49.940890] Got packet of 234 bytes
[2017-06-27 10:13:49.941016] Got packet of 138 bytes
[2017-06-27 10:13:49.941071] Got packet of 234 bytes
[2017-06-27 10:13:49.941183] Got packet of 138 bytes
[2017-06-27 10:13:49.941234] Got packet of 186 bytes
[2017-06-27 10:13:49.941367] Got packet of 114 bytes
[2017-06-27 10:13:49.980465] Got packet of 66 bytes
[2017-06-27 10:13:49.984109] Got packet of 214 bytes
[2017-06-27 10:13:50.94819] Got packet of 60 bytes
[2017-06-27 10:13:50.97228] Got packet of 92 bytes
[2017-06-27 10:13:50.130323] Got packet of 455 bytes
[2017-06-27 10:13:50.130534] Got packet of 455 bytes
[2017-06-27 10:13:50.199395] Got packet of 214 bytes
^C
Finishing the capture... that's all folks!
root@itaqua:/lab/users/Miguel/miguel/disciplinas/progredes/libpcap#
```

Função Principal

```
#include <iostream>

#include "pcapwrapper.h"

using namespace std;

int main (int argc, char *argv[]) {

        PcapWrapper pcapwrapper (argv [1]);

        //pcapwrapper.setFilter ("port 80");
        pcapwrapper.run ();

        return 0;
}
```

Classe PcapWrapper (*.h)

```
#include <iostream>
#include <stdlib.h>
#include <pcap.h>
#include <signal.h>
#include <unistd.h>

using namespace std;

/*! \class PcapWrapper
    \brief This class encapsulates some functions of libpcap.

    This class can be used as an exercise for a simple packet capture.
*/
class PcapWrapper {

        public:

        PcapWrapper (char * = NULL);
        ~PcapWrapper ();

        void run (int = 0) ;

        void setFilter (char *);

        private:
                char * dev, * filter;
                char errbuf [PCAP_ERRBUF_SIZE];
                bpf_u_int32 mask, net;
                struct bpf_program fp;
                pcap_t * handle;
};
```

```
#include "pcapwrapper.h"

pcap_t * _handle;

void terminate_process (int) {
        pcap_breakloop (_handle);
}

void got_packet(u_char *args, const struct pcap_pkthdr *header, const u_char *packet) {
        time_t nowtime;
        struct tm *nowtm;
        char tmbuf [64];

        nowtime = header->ts.tv_sec;
        nowtm = localtime(&nowtime);
        strftime(tmbuf, sizeof tmbuf, "%Y-%m-%d %T", nowtm);

        cout << "[" << tmbuf << "." << header->ts.tv_usec << "]" << " Got packet of " << header->caplen << " bytes" << endl;
}


PcapWrapper::PcapWrapper (char * filter_) {

        dev = pcap_lookupdev (errbuf);

        if (dev == NULL) {
                cerr << "Couldn't find default device: " << errbuf << endl;
                cerr << "*** You must probably run as root! ***" << endl;
                exit (EXIT_FAILURE);
        }

        cout << "Device: " << dev << endl;

        // Find the properties for the device
        if (pcap_lookupnet (dev, &net, &mask, errbuf) == -1) {
                cerr << "Couldn't get netmask for device" << dev << ":" << errbuf;
                net = 0;
                mask = 0;
        }

        handle = pcap_open_live (dev, BUFSIZ, 1, 0, errbuf);
        if (handle == NULL) {
                cerr << "Couldn't open device " << dev << " : " << errbuf << endl;
                exit (EXIT_FAILURE);
        }

        if (filter_ != NULL)
                setFilter (filter);
}
```

Classe PcapWrapper (*.cpp)

```
PcapWrapper::~PcapWrapper () {
        cout << "\nFinishing the capture... that's all folks!" << endl;

        if (filter != NULL) pcap_freecode(&fp);
        pcap_close (handle);
}

void PcapWrapper::run (int stoptime_) {
        // stop after stoptime seconds
        if (stoptime_ > 0) {
                cout << "*** Finishing after " << stoptime_ << " seconds ***" << endl;
                signal (SIGALRM, terminate_process);
                alarm (stoptime_);
                _handle = handle;
        } else {
                cout << "*** Press ctrl-c to finish ***" << endl;
                signal (SIGINT, terminate_process);
                _handle = handle;
        }

        // now we can set our callback function
        pcap_loop (handle, -1, got_packet, NULL);
}

void PcapWrapper::setFilter (char * filter_) {
        filter = filter_;

        // Compile and apply the filter
        if (pcap_compile (handle, &fp, filter_, 0, net) == -1) {
                cerr << "Couldn't parse filter " << filter_ << ":" << pcap_geterr (handle);
                exit (EXIT_FAILURE);
        }

        if (pcap_setfilter (handle, &fp) == -1) {
                cerr << "Couldn't parse filter " << filter_ << ":" << pcap_geterr (handle);
                exit (EXIT_FAILURE);
        }
}
```

Classe PcapWrapper (*.cpp)

Makefile

```
CPP = g++
CPPFLAGS = -Wall
LD = g++
LIBS = -lpcap

PROGRAM = captura
OBJS = main.o pcapwrapper.o

all: $(PROGRAM)

$(PROGRAM): $(OBJS)
	$(LD) $(OBJS) $(LIBS) -o $@

.cpp.o:
	$(CPP) $(CPPFLAGS) $(LIBS) -c $<

clean:
	rm -f *.o $(PROGRAM)
```

Documentação

- Uso do **Doxygen**:
  - Programa que facilita a documentação do código com vários níveis de detalhes
    - Instalação via apt-get:
      - **`apt-get install doxygen`**
    - Documentação disponível em:
      - http://www.stack.nl/~dimitri/doxygen/
    - Entre outros formatos, pode gerar documentos em:
      - HTML [captura docs html] : Abrir o arquivo html/index.html
      - PDF: [captura docs pdf]: Abrir o arquivo refman.pdf

Uso Básico do Doxygen

- Criação do arquivo de configuração (**Doxyfile**):

```
$ doxygen -g
```

- Exemplo de configuração no **Doxyfile**:

```
PROJECT_NAME           = "Capturing with libpcap"
EXTRACT_ALL            = YES
EXTRACT_PRIVATE        = YES
HAVE_DOT               = YES
UML_LOOK               = YES
GENERATE_TREEVIEW      = YES
INPUT                  = main.cpp pcapwrapper.h pcapwrapper.cpp
```

- Geração da documentação:

```
$ doxygen Doxyfile
```

- Geração do PDF:

```
$ cd latex; make
```

Documentação Gerada

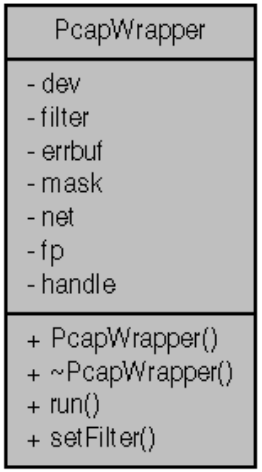
Class Documentation

3.1 PcapWrapper Class Reference

This class encapsulates some functions of libpcap.

`#include <pcapwrapper.h>`

Collaboration diagram for PcapWrapper:



Public Member Functions

- PcapWrapper (char *=NULL)
- ~PcapWrapper ()
- void run (int=0)
- void setFilter (char *)

Private Attributes

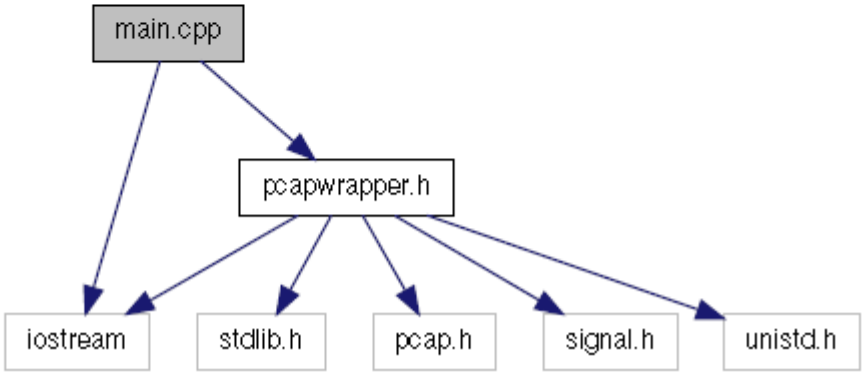
- char * dev
- char * filter
- char errbuf [PCAP_ERRBUF_SIZE]
- bpf_u_int32 mask
- bpf_u_int32 net
- struct bpf_program fp
- pcap_t * handle

Documentação Gerada

4.1 main.cpp File Reference

```
#include <iostream>
#include "pcapwrapper.h"
```

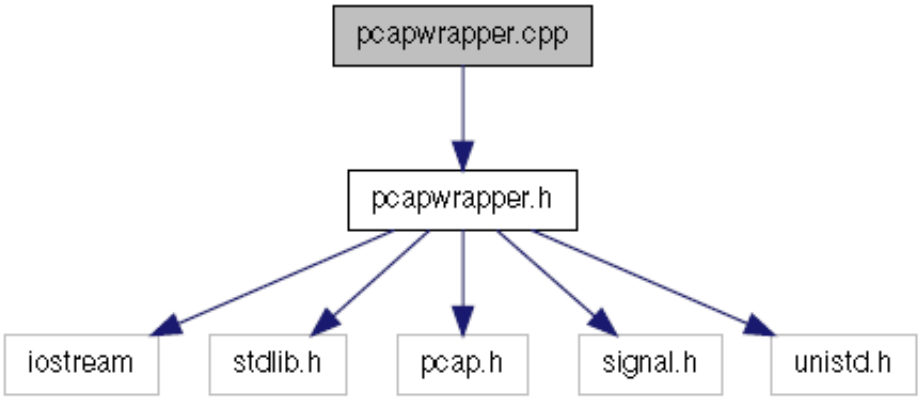
Include dependency graph for main.cpp:



Documentação Gerada

```
#include "pcapwrapper.h"
```

Include dependency graph for pcapwrapper.cpp:



Tem isso tudo e muito mais nos documentos gerados!

Leitura Recomendada

- Brian Gough, "*An Introduction to GCC*", 2nd edition, **Network Theory Ltd**, 2005
- "GNU `make'", disponível em **http://www.gnu.org/software/make/manual/make.html**
- "GNU Automake", disponível em **http://sources.redhat.com/automake/automake.html**
- "Static, Shared Dynamic and Loadable Linux Libraries", disponível em **http://www.yolinux.com/TUTORIALS/LibraryArchives-StaticAndDynamic.html**
- Doxygen, Disponível em http://www.stack.nl/~dimitri/doxygen/