Orientação de estudo em programação C – 2007

Mathias JK Erdtmann

8 de julho de 2007

Sumário

1 Instruções iniciais

Existiu um tempo onde um compilador C custava o preço de um caça supersônico. Existiu um tempo onde um compilador C custava o preço de um carro. Existiu um tempo onde um compilador C custava o preço de um HD de 5MB (daquele tempo).

Hoje um compilador C custa $0,00. Hoje um sistema operacional custa $0,00. A informação tornou-se barata e muitas pessoas se esforçaram para fornecer ao mundo aquilo que se acreditou ser artigo de necessidade básica. O movimento GNU que gerou a onda de desenvolvimento colaborativo e aberto nos proporciona as melhores ferramentas de desenvolvimento do mercado de forma livre, sendo que o que separa as melhores opções do ponto de vista técnico e o líder do mercado são somente configurações circunstanciais.

Como as decisões não devem ser tomadas não somente com avaliações circunstanciais, mas também técnicas, opta-se aqui por opções tecnicamente superiores que não agridam/violem/deixem de funcionar com as configurações atuais. Em resumo, o que quer-se deixar claro é que as opção por ferramentas livres [1] devem manter a compatibilidade com os requisitos de transportabilidade (por exemplo, em sistemas MS-Windows e GNU/Linux) e usabilidade.

Estas instruções iniciais visam ajudar a preparação do ambiente de trabalho e, posteriormente, dar uma visão geral sobre a linguagem de programação C. É interessante que se siga mais ou menos esta ordem, de forma que ao experimentar a sintaxe/semântica da linguagem C, já se possua o ambiente pronto para testes.

1.1 Compilador C

Para criar executáveis a partir do código C, vai ser necessário de um compilador C. A melhor opção atualmente é o gcc (GNU Compiler Collection), pois apresenta:

- liberdade: de distribuição, modificação, re-distribuição, comercialização.
- preço imbatível: $ 0,00.
- busca constante pela padronização ANSI e POSIX.
- alta qualidade de binários gerados.

[1] a justificativa para superioridade de ferramentas livres fica para outro documento

- transportabilidade: roda e compila para 42 plataformas (diferentes microcontroladores) e um variado número de sistemas operacionais.

Entre os sistemas operacionais nos quais o gcc pode ser utilizado, destacam-se:

- todos os MS-Windows
- GNU/linux
- *BSD

O gcc já é considerado o compilador padrão para sistemas UNIX.
`http://en.wikipedia.org/wiki/GNU_Compiler_Collection`

1.2 Instalando o gcc

1.2.1 GNU/Linux

Deve usar-se o gerenciador de pacotes da distribuição. No Ubuntu, por exemplo, deve-se usar o synaptic para garantir que o pacote build-essential esteja instalado. Isto pode ser feito por apt diretamente, usando o comando:

```
$ sudo apt-get install build-essential
```

A instalação padrão do Ubuntu já inclui o gcc e o editor de textos gedit que pode ser utilizado para editar os fontes de programas C com bastante facilidade. Caso sejam desejadas características especiais, é possível configurar bastante coisa nas preferências e ainda tem-se a opção de instalar os plugins extra do gedit (pacote gedit-plugins):

```
$ sudo apt-get install gedit-plugins
```

Outras opções disponíveis são: emacs, kate, kdevelop, anjuta, todas elas instaláveis a partir do repositório padrão da maioria das distribuições. Além destas, destaca-se ainda o IDE Eclipse, que na versão Europa inclui um empacotamente para desenvilvimento C/C++ completo (ainda assim é necessário instalar o pacote build-essential). O Eclipse é discutido em mais detalhes na seção sobre MS-Windows 1.2.2 deste documento (o IDE é transportável) e pode ser encontrado na página:

`http://www.eclipse.org/downloads/`

1.2.2 MS-Windows

Sistemas MS-Windows não foram projetados com o foco em ambientes de desenvolvimento, de forma que considerável esforço precisa ser empreendido para deixá-lo pronto para o trabalho.

Opção1 - MinGW

A instalação do MinGW é um tanto confusa, mas pode-se usar outros pacotes para instalação. O pacote mais famoso que incluí as ferramentas do MinGW é o Dev-C++, que além do(s) compiladores C, C++ do MinGW, inclui uma IDE completa.

`http://www.bloodshed.net/devcpp.html`

Opção2 - cygwin

A idéia do cygwin é fornecer um ambiente parecido com o UNIX em windows. Inclui, além do gcc, muitos outros pacotes e um ambiente de linha de comando baseado no bash. Obtém-se a instalação padrão no sítio do cygwin e pode-se escolher os pacotes durante o processo de instalação via rede.

`http://www.cygwin.com/`

Caso instale o gcc via cygwin (ou ainda via MinGW), é desejável um editor de textos que forneça facilidades para a programação. Novamente existem muitas opções. A mais recomendável é o Eclipse/CDT.

O IDE Eclipse (famoso pelo ambiente de desenvolvimento Java que proporciona) pode ser dotado com o adicional CDT para desenvolvimento C. Na versão Europa, o Eclipse conta com um empacotamento especial para quem quer somente desenvolver código C/C++. O Eclipse Europa pode ser encontrado no endereço:

`http://www.eclipse.org/downloads/`

O Eclipse precisa do ambiente de execução Java para rodar e o pacote distribuído não inclui o compilador C, precisando então o mesmo ser instalado separadamente, com algum dos métodos citados acima. Instruções para união do Eclipse ao gcc do cygwin podem ser encontradas em:

`http://eclipsewiki.editme.com/UsingCDTWithCygwin`

Opção drástica - VirtualBox

A opção drástica é a execução de uma máquina virtual para poder executar algum sistema GNU/Linux a partir do prórpio MS-Windows. Após a instalação da máquina virtual, é questão de instalar configurar o ambiente de programação como visto na seção 1.2.1. Um bom

virtualizador pode ser encontrado em:

`http://www.virtualbox.org/`

Opção mais drástica - Ubuntu

A opção mais drástica inclui a instalação do sistema operacional Ubuntu. Após este passo, refira-se à seção 1.2.1 para prosseguir a instalação do ambiente de desenvolvimento C.

Esta opção é a que geralmente produz melhores resultados, mas se for realmente necessário produzir e testar os programas para a plataforma MS-Windows, pode acabar sendo uma opção um tanto quanto trabalhosa (assim como na utilização de máquinas virtuais).

1.3 Código C

O artigo em inglês da wikipedia é uma boa introdução à linguagem:

`http://en.wikipedia.org/wiki/C_(programming_language)`

Mas para uma primeira leitura simplificada, recomenda-se ignorar os temas: "pointers, memory management, operator precedence"(mas não ignorar o ponteiro para o artigo "*Operators in C and C++*")

Após a leitura do artigo (e de possivelmente 3 diferentes artigos que foram abertos ao longo da leitura usando os ponteiros[2]), recomenda-se a leitura do arquivo `lista1.txt` (anexo) contendo exercícios e tarefas para ter uma idéia do desafio que se espera para esta primeira etapa (ler não significa resolver).

Retornando então ao artigo da wikipedia, no final do mesmo podem ser encontrados alguns ponteiros interessantes de tutoriais C. Estes tutoriais muitas vezes começam de forma simples, mas podem acabar sendo bastante complexos. Neste contexto, deve-se lembrar dos exercícios e tarefas que deseja-se cumprir e ler apenas 50% além do necessário para conseguir resolver os problemas. Neste momento inicial de aprendizado, os ponteiros particularmente mais interessantes são (em ordem):

Howstuffworks C Programming

Learning to C

The C Book

A estrutura de programas C que deve-se ter bastante clara é a seguinte:

[2]este processo pode ser chamado de hiperleitura, e abrange também os outros 9 artigos lidos decorrentes de 3 artigos abertos usando referências de 3 artigos abertos, e assim sucessivamente...

```
//-----------------------------------
//diretiva de pré-processamento, que "cola" o conteúdo do
// arquivo "stdio.h" neste local:
#include <stdio.h>

//declaração de função e (em especial) da função "main",
// identificada pelo montador como a função a ser associada
// à execução do programa:
int main()
{
//declaração de variáveis, com o intuito de armazenar o
// setor de memória delas. Em C (padrão 1990) é necessário
// declarar todas as variáveis no começo da função, e elas
// vão existir durante toda a execução da funçao (mas não
// fora dela):
    int a;
    int b;
    int c;
    float d;


    a = 5;
    b = 7;
    c = a + b;
    d = (float)c;

//as funções printf estão declaradas no arquivo "stdio.h",
// e são o motivo pelo qual teve-se que incluir o arquivo
// no início do código:
    printf("%d + %d = %d\n", a, b, c);
    printf("%d + %d = %f\n", a, b, d);

    return 0;
}

//-EOF-------------------------------
```

Com estas orientações espera-se que seja possível resolver os problemas básicos que possam aparecer. Bom divertimento!

A lista1.txt

```
Exercícios de C
===============

Lista 1
-------

-= LEIA *TODAS* AS DICAS GERAIS ANTES DE COMEÇAR =-


DICAS GERAIS:
- O pacote "build-essential" deve ser instalado no Ubuntu para poder compilar após instala
- Pode-se compilar um arquivo "programa.c" (quando não se usa bibliotecas extra) simplesmen
$ make programa
- Pode-se compilar um arquivo "programa.c" (quando não se usa bibliotecas extra) simplesmen
$ gcc programa.c -o programa
- Para executar um programa que não está no path do sistema, é necessário indicar o caminho
$ ./programa
- STDIN é a abreviação para entrada padrão, ou seja, teclado no terminal.
- STDOUT é a abreviação para saída padrão, ou seja, tela do terminal.
- Pode-se usar redirecionamento de arquivo para STDIN e STDOUT, para poder guardar as entra
$ cat entrada.txt > ./executavel > saida.txt
- Pode-se ler um arquivo no terminal usando
$ cat queroLerEste.txt

- Entrada e saída devem ser estritamente obedecidas.
- A solução mais simples é a mais correta.
- Assume-se que qualquer array de menos de 1MB é pequeno e não vale a pena se preocupar com
- Todo o enunciado (apesar de longo) deve ser lido, prestando atenção aos detalhes.


-= EXERCÍCIOS =-

1. Crie um programa que reimplemente a funcionalidade do comando "echo" (tudo que lê pelo S

dicas de procura: scanf, printf

2. Crie um programa que reimplemente o "cat", retornando no STDOUT o conteúdo de um arquivo

dicas de procura: argc, argv, fopen, fscanf

3. Crie um programa que conte o número de palavras em um arquivo cujo nome é passado por li

dicas de procura: while, EOF C

4. Crie um programa que receba uma linha de texto pelo STDIN, substitua todos os pontos fin

dicas de procura: ASCII table, gets, char array
```

-= TAREFAS =-

(as tarefas são consideravelmente mais difíceis que os exercícios, mas deve-se tentar fazer

TAREFA1. Detecção de início/fim de sinal

Tem-se um vetor de tamanho M (M<512), contendo uma seqüência de inteiros representativos de

O programa (C) deve receber pelo STDIN o valor de M na primeira linha, seguido do valor típ

O programa deve finalizar ao receber -1 como tamanho M.

EXEMPLO:

ENTRADA:
20
0
-1
2
-2
1
2
3
-1
-1
6
9
10
5
2
-2
-5
-4
-2
-3
-2
0

10
100
102
103
99
98
97
100
101
102
103

```
99

-1

SAÍDA:
11
0
6
9
10
5
2
-2
-5
-4
-2
-3
-2
0
100
------------------------------

dicas para procura: <nada novo>


TAREFA2. Ordenação (mais ou menos) fácil
----------------------------------------

Nome do fonte:  sortKindaEasy.c
Nome do executável: sortKindaEasy

Recebendo um conjunto de N inteiros (com N<8192), deseja-se ordená-los pela ordem crescente

O programa (C) deve receber no STDIN o valor de N na primeira linha, indicando quantos será

O programa deve repetir este procedimento de entrada/saída até que seja recebido o valor "-

EXEMPLO:
----------------------------------------
ENTRADA:
10
1
2
3
4
5
6
7
8
9
```

```
0

4
1232354
13431612
4615123
763465321

-1

SAÍDA:
10
0
1
2
3
4
5
6
7
8
9
4
763465321
13431612
4615123
1232354
------------------------------

dicas para procura: bucket sort, qsort, quick sort, int array, remainder operator


TAREFA3. Mostrando informações sobre o usuário.
-----------------------------------------------

Nome do fonte:  userinfo.c
Nome do executável: userinfo
Arquivos extra: passwd

Um arquivo mantendo informações sobre os usuários de um sistema UNIX tem o seguinte aspecto

- cada linha contém informações sobre um usuário
- campos de informação são separados por ":"
- os campos são ordenados em:

ULOGIN:PASSWD:UID:GID:FULLNAME:HOMEDIR:SHELL

sendo que:
-- ULOGIN é o nome do usuário para logar no sistema
-- PASSWD é a senha ("x" significa que ela é armazenada em /etc/shadow)
```

```
-- UID é o número que identifica o usuário
-- GID é o número que identifica o grupo (principal) do usuário
-- FULLNAME é o nome completo, podendo conter múltiplas palavras
-- HOMEDIR é a pasta no sistema de arquivos para o uso pessoal do usuário
-- SHELL é o comando a ser executado assim que o usuário entrar no sistema, para execução d

A tarefa consiste em criar um programa (C) que tenha o seguinte comportamento:

- Receber UID ou ULOGIN pelo STDIN.
- Retornar no STDOUT "UID:" seguido do UID, " ULOGIN:" seguido do ULOGIN, " FULLNAME:" segu
  UID, ULOGIN e FULLNAME devem ser coerentes com o que é lido no arquivo "/etc/passwd" ou c
  Assume-se que nenhum ULOGIN é maior que 255 caracteres e nenhum FULLNAME é maior que 4096

- O programa deve repetir este procedimento de entrada/saída até que seja recebido o valor

EXEMPLO (para arquivo anexo):
-------------------------------------
ENTRADA:
mathias
65534
mefistofeles
-1

SAÍDA:
UID:1000 ULOGIN:mathias FULLNAME:Mathias JK Erdtmann
UID:65534 ULOGIN:nobody FULLNAME:nobody
UID:-1 ULOGIN:notfound FULLNAME:User not found
-------------------------------------

Arquivos:
--FILE:passwd-----------------------
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
dhcp:x:100:101::/nonexistent:/bin/false
```

```
syslog:x:101:102::/home/syslog:/bin/false
klog:x:102:103::/home/klog:/bin/false
messagebus:x:103:104::/var/run/dbus:/bin/false
avahi-autoipd:x:104:108:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
avahi:x:105:109:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
cupsys:x:106:111::/home/cupsys:/bin/false
haldaemon:x:107:112:Hardware abstraction layer,,,:/home/haldaemon:/bin/false
hplip:x:108:7:HPLIP system user,,,:/var/run/hplip:/bin/false
gdm:x:109:116:Gnome Display Manager:/var/lib/gdm:/bin/false
mathias:x:1000:1000:Mathias JK Erdtmann,,,:/home/erdtmann:/bin/bash
beagleindex:x:110:65534::/var/cache/beagle:/bin/false
sshd:x:111:65534::/var/run/sshd:/usr/sbin/nologin
ntp:x:112:119::/home/ntp:/bin/false
cristiane:x:1001:1002:cristiane,,,,:/home/cristiane:/bin/bash
mysql:x:113:121:MySQL Server,,,:/var/lib/mysql:/bin/false
terp:x:114:65534::/home/terp:/bin/false
saned:x:115:122::/home/saned:/bin/false
------------------EOF--------------------
```

dicas para procura: /etc/passwd, atoi