

Algoritmos

Introdução a Computação

Prof. Hitoshi Nagano, Ph.D.

Aula 4

Primeiro Programa em C

Demonstração


Escrever, compilar e rodar o primeiro programa em C:

```
#include <stdio.h>
int main()
{
    printf("ola mundo!\n");
    return 0;
}
```


Anatomia

```
#include <stdio.h>
int main()
{
    printf("ola mundo!\n");
    return 0;
}
```


Anatomia

```
#include <stdio.h>  
 int main()  
{  
  
    printf("ola mundo!\n");  
    return 0;  
  
}
```


Anatomia

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

```
int main()
```



```
{
```

```
    printf("ola mundo!\n");
```



```
    return 0;
```



```
}
```


Anatomia

```
#include <stdio.h>
int main()
{
    printf("ola mundo!\n");
    return 0;
}
```



Anatomia

```
#include <stdio.h>  
int main()  
{
```



```
printf("ola mundo!\n");  
return 0;
```

```
}
```


Anatomia

```
#include <stdio.h>  
int main()  
{
```




```
printf("ola mundo!\n");  
return 0;
```

```
}
```



Anatomia

```
#include <stdio.h>
int main()
{
    printf("ola mundo!\n");
    return 0;
}
```

Two green arrows pointing downwards. The first arrow points to the opening parenthesis of the printf function call. The second arrow points to the semicolon at the end of the return statement.


Anatomia

```
#include <stdio.h>
int main()
{
    printf("ola mundo!\n");
    return 0;
}
```




Anatomia

```
#include <stdio.h>
int main()
{
    printf("ola mundo!\n");
    return 0;
}
```



Anatomia

```
#include <stdio.h>
int main()
{
    printf("ola mundo!\n");
    return 0;
}
```




Anatomia

- os principais componentes -

```
#include <stdio.h>
int main()
{
    printf("ola mundo!\n");
    return 0;
}
```


Anatomia




```
#include <stdio.h>
int main()
{

    printf("ola mundo!\n");
    return 0;

}
```


Anatomia

```
#include <stdio.h>  
 int main()  
{  
  
    printf("ola mundo!\n");  
    return 0;  
  
}
```


Anatomia

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

```
int main()
```

```
{
```

```
    printf("ola mundo!\n");
```



```
    return 0;
```

```
}
```


~~Anatomia?~~

Sintaxe

Linux: 3 comandos básicos

para criar, compilar e rodar programas

- `gedit`
- `clang` ou `gcc`
- `./<nome do executavel>`

Linux: 3 comandos básicos

para criar, compilar e rodar programas

EXEMPLO:

```
$ gedit hello.c
```

```
$ gcc hello.c -o hello
```

```
$ ./hello
```


Demonstração

- Comandos

```
$ gedit helloworld.c
```

```
$ gcc helloworld.c -o olamundo
```

```
$ ./olamundo
```


Jogo dos 7 erros

Versão correta

```
#include <stdio.h>
int main()
{
    printf("ola mundo!\n");
    return 0;
}
```


ache o erro - 1

```
#include <stdio.h>
int main()
{
    printf("ola mundo!\n");
    return 0;
}
```


ache o erro - 2

```
#include <stdio.h>
int main()
{
    printf("ola mundo!\n")
    return 0;
}
```


ache o erro - 3

```
#include <stdio.h>
int main()
{
    printf("ola mundo!\n");
    return 0;
}
```


ache o erro - 4

```
#include <stdio.h>
int main()
{
    printf("ola mundo!\n");
    return 0;
}
```


ache o erro - 5

```
$ gcc hello.c -o hello
```


ache o erro - 6

```
$ gedit hello
```


ache o erro - 7

```
$ ./hello.c
```


Bugs

Bugs...

- Erros de compilação (*compilation error*)
- Erros de execução (*runtime error*)
- Erros de lógica (*logic error*)

Bugs...

- Erros de compilação (*compilation error*)

```
int a = 5 + ; printf("Hello\n");
```

- Erros de execução (*runtime error*)

```
int a = 0; int b = 5/a;
```

- Erros de lógica (*logic error*)

```
int a=2, b=2; printf("a+b=%d", a-b)
```


Bugs...

- Esquecer `{...}`, ou esquecer `(...)`, ou esquecer `"..."`, ou esquecer `[...]`
Esquecer de fechar com `...}`, `...)`, `..."`, `...]`
Confundir parentesis com chaves, Exemplo: `{...)`, ou `(...]`
- Esquecer `“;”` ao final de cada instrução
- Esquecer de incluir bibliotecas, digitar errado as bibliotecas.
Por exemplo: esquecer `#include <stdio.h>`
Por exemplo: digitar `#include <studio.h>`
- Esquecer de declarar variáveis.
Por exemplo esquecer: `int a = 0;`
- Esquecer de inicializar variáveis.
Por exemplo esquecer: `a = 0;`

Bugs...

- Conteúdo entre chaves {...}. Ex: dentro do loop
 - quais instruções precisam estar dentro das chaves
 - quais instruções precisam estar fora
 - quais instruções podem estar fora
- Conteúdo entre chaves {...}. Ex: condições
 - quais depois do if
 - quais depois do else
 - quais fora da estrutura if/else

bugs lógicos... normalmente compila, ... mas resultados errados

gotofail

. . .

```
hashOut.data = hashes + SSL_MD5_DIGEST_LEN;
hashOut.length = SSL_SHA1_DIGEST_LEN;

if ((err = SSLFreeBuffer(&hashCtx)) != 0)
    goto fail;
if ((err = ReadyHash(&SSLHashSHA1, &hashCtx)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &clientRandom)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
    goto fail;
    goto fail;
if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
    goto fail;

err = sslRawVerify(...);
. . .
```

<https://nakedsecurity.sophos.com/2014/02/24/anatomy-of-a-goto-fail-apples-ssl-bug-explained-plus-an-unofficial-patch/>

mais bugs lógicos

- while (condição) {...}
condição é do tipo: enquanto verdadeira → execute {...}
(diferente do Scratch: que é do tipo repita até a condição se tornar verdadeira)
- Exemplo: viagem de carro para Bauru com criança de 10 anos

Em C:

```
while (ainda na estrada?)  
{  
    falar “quanto tempo falta?”;  
    esperar (10 mins);  
}
```

Em Scratch:

```
repeat until (chegou em Bauru?)  
{  
    falar “quanto tempo falta?”;  
    esperar (10 mins);  
}
```


bugs lógicos++

- esquecer de incrementar a variável de contagem, como tempo ou contador.
- atenção com os casos limites (*corner cases*), por exemplo, o primeiro e último casos de um loop.

bugs lógicos... normalmente compila, ... mas resultados errados

bugs lógicos++

- *fencepost error*

Para 5 lances de cabos,
quantos postes precisam?

Resposta: 5 ~~postes~~

6 postes



bugs lógicos... normalmente compila, ... mas resultados errados

bugs lógicos++

- *fencepost error*

Quantos números são exibidos?

```
for (int i = 5; i >=0; i--)  
    printf("i = %d\n", i);
```



bugs lógicos... normalmente compila, ... mas resultados errados

Demonstração (II)

Escolher um problema da série de exercícios I
e implementar em C.

Feedback anonimo

<http://sayat.me/UnisantosAlgoritmos>

Problemas de Imprecisão

Problemas

- transbordo de inteiro
integer overflow
- imprecisão do ponto flutuante
floating point imprecision