

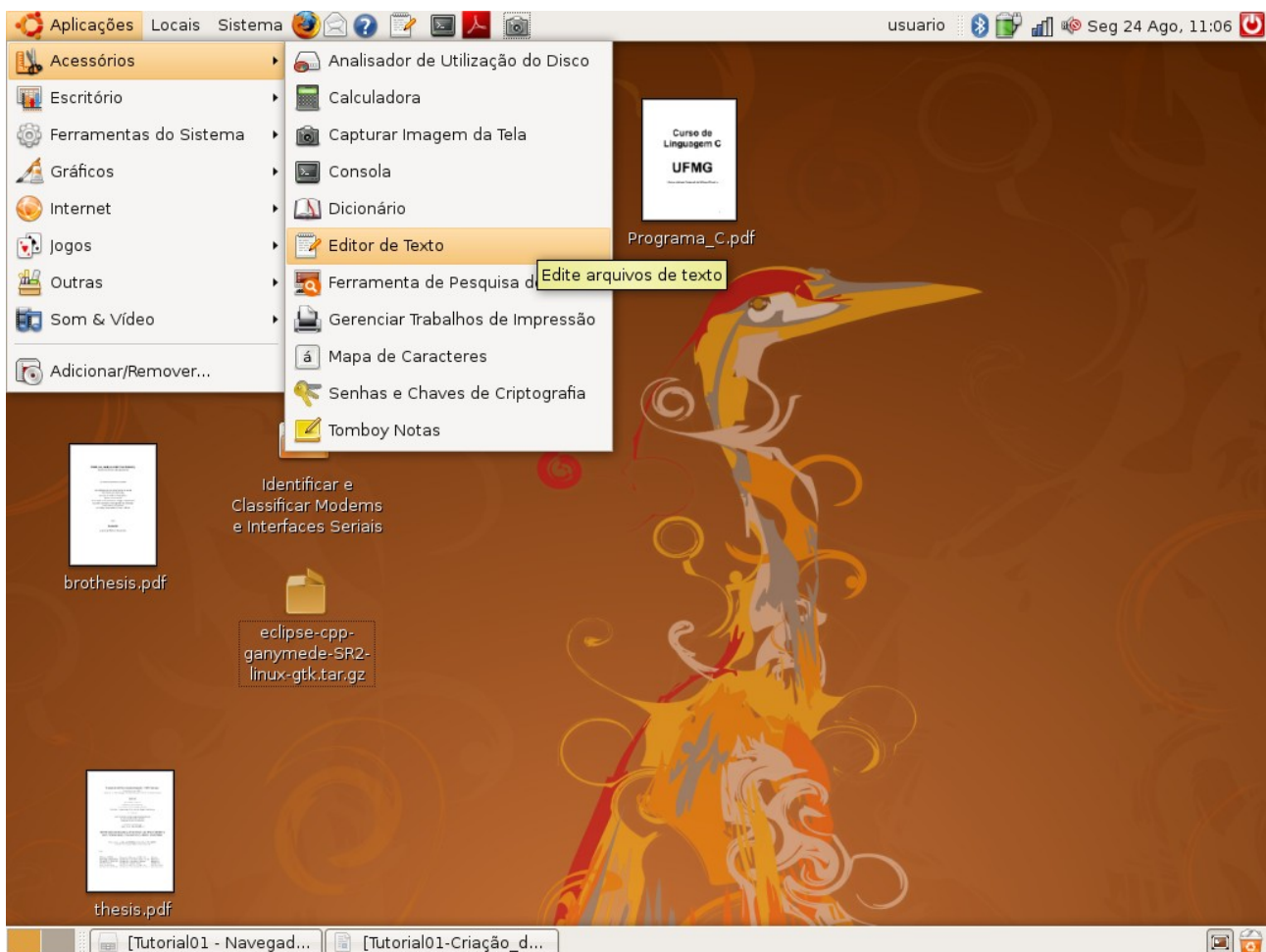
**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA – IFCE**  
**ETEC - CURSO TÉCNICO EM INFORMÁTICA**  
**LÓGICA DE PROGRAMAÇÃO – 2009.2**

**PROF. Jean Marcelo da Silva Maciel**

**TUTORIAL 01 – Criação do primeiro programa no GNU/Linux Ubuntu, utilizando o compilador gcc.**

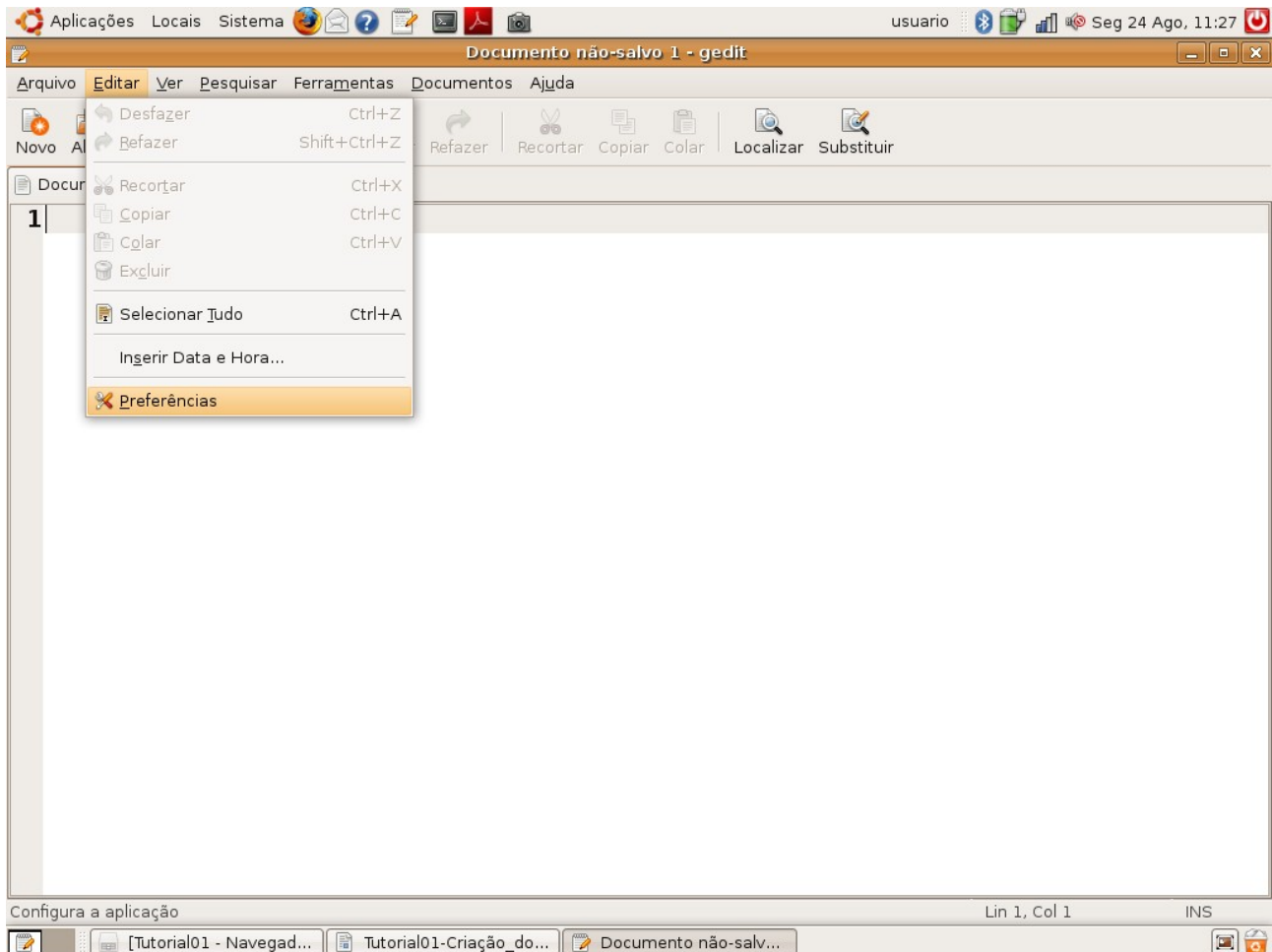
**Objetivo:** Neste tutorial vamos aprender a criar um programa fonte (código fonte) utilizando um editor de texto, no caso, o **gedit** do Ubuntu. Após a criação do programa fonte, vamos compilar o programa para geração de um programa executável, utilizando para tal o compilador **gcc**, que é o compilador C padrão de sistemas GNU/Linux. Ao final, mas sem a intenção de esgotar todas as possibilidades de erros, serão apresentados alguns erros possíveis de acontecer durante o processo de compilação.

**Passo 1:** No sistema operacional Ubuntu abra o editor de textos **gedit**. Vá em **Aplicações/Acessórios/Editor de Textos**, conforme figura abaixo:

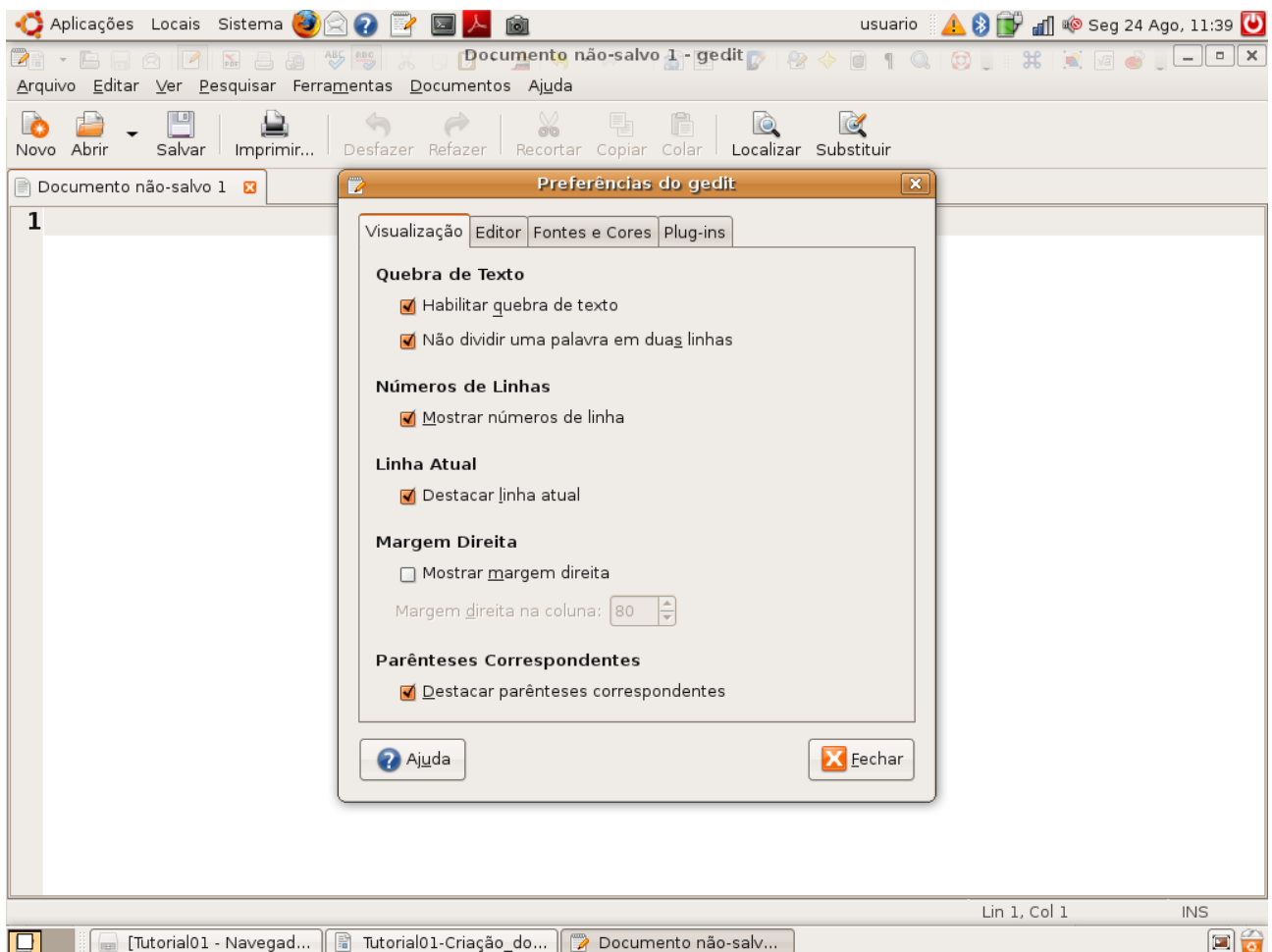


**Abrindo o editor gedit**

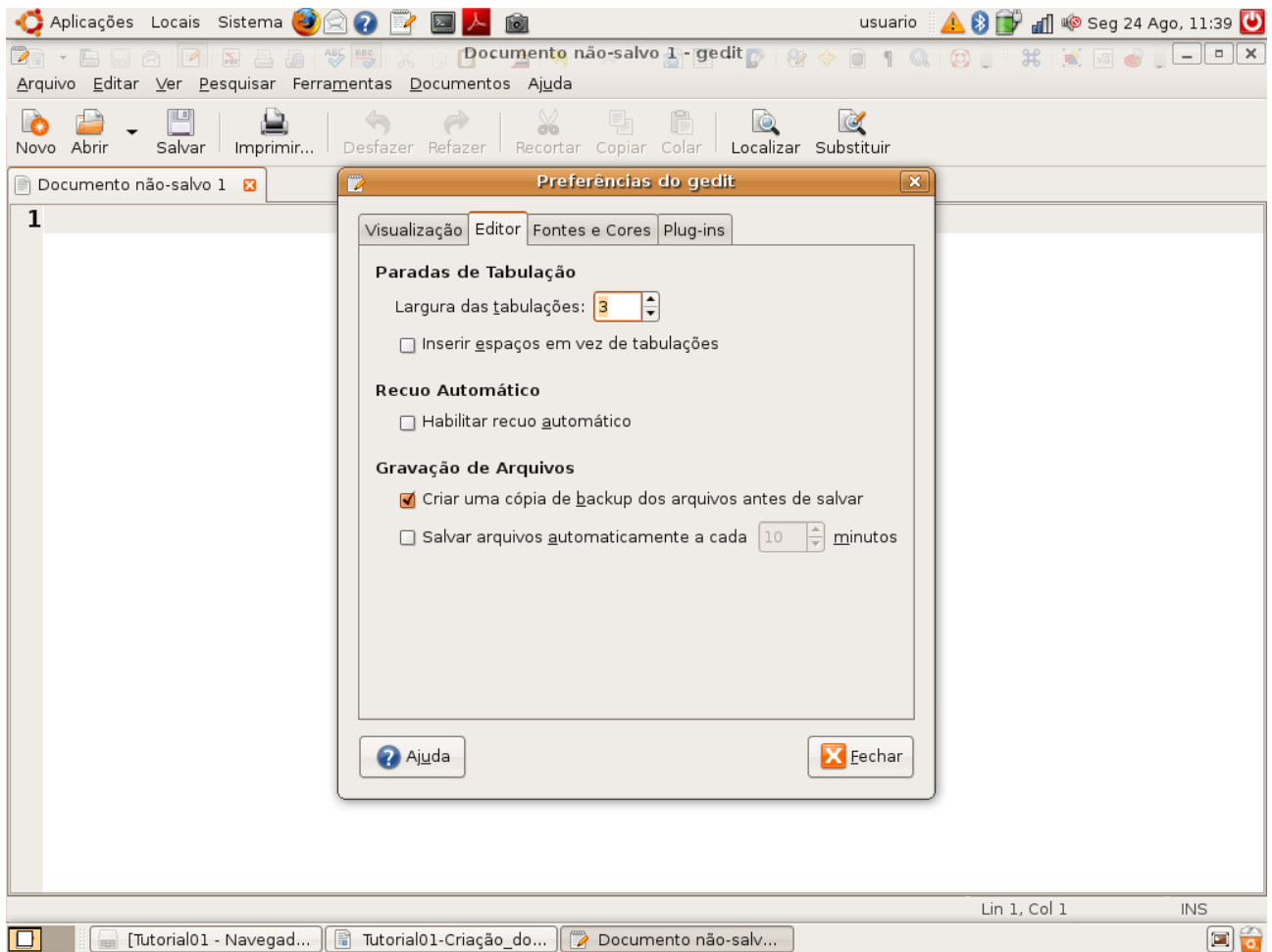
**Passo 2:** Configure o gedit para mostrar o número das linhas, *Editar/Preferências/Visualização/Número de linhas*, e depois configure a tabulação para 3, *Editar/Preferências/Visualização/Editor/Largura das tabulações*. Veja as figuras abaixo. Após essas configurações, o número de cada linha será mostrado e sempre que você teclar <tab> serão dados três espaços. Utilize a tecla <tab> para **indentar** o seu programa fonte, de forma que o mesmo fique mais legível, tanto para você como outros programadores. Observação: você precisa fazer este passo somente uma vez, pois, após estar configurado o gedit, as configurações permanecem para edições de outros programas fonte.



**Selecionar Preferências do gedit.**



**Na aba *Visualização*, seleccionar *Mostrar números de linhas*.**



**Na aba *Editor*, configurar *Largura de tabulações* para 3.**

**Passo 3:** Edite o seu primeiro programa fonte, conforme abaixo:

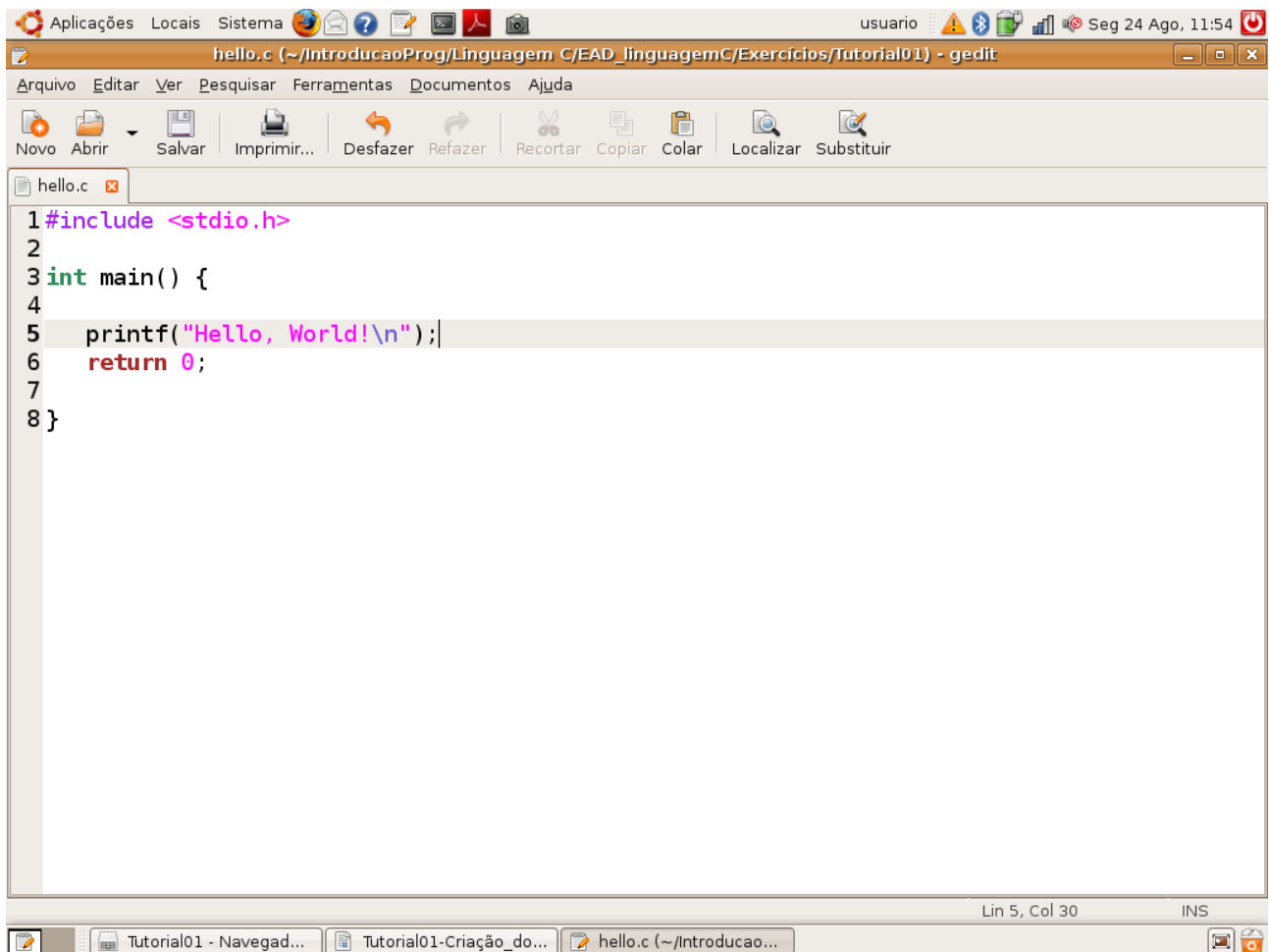
```
#include <stdio.h>
```

```
int main() {
```

```
    printf("Hello, World!\n");
```

```
    return 0;
```

```
}
```

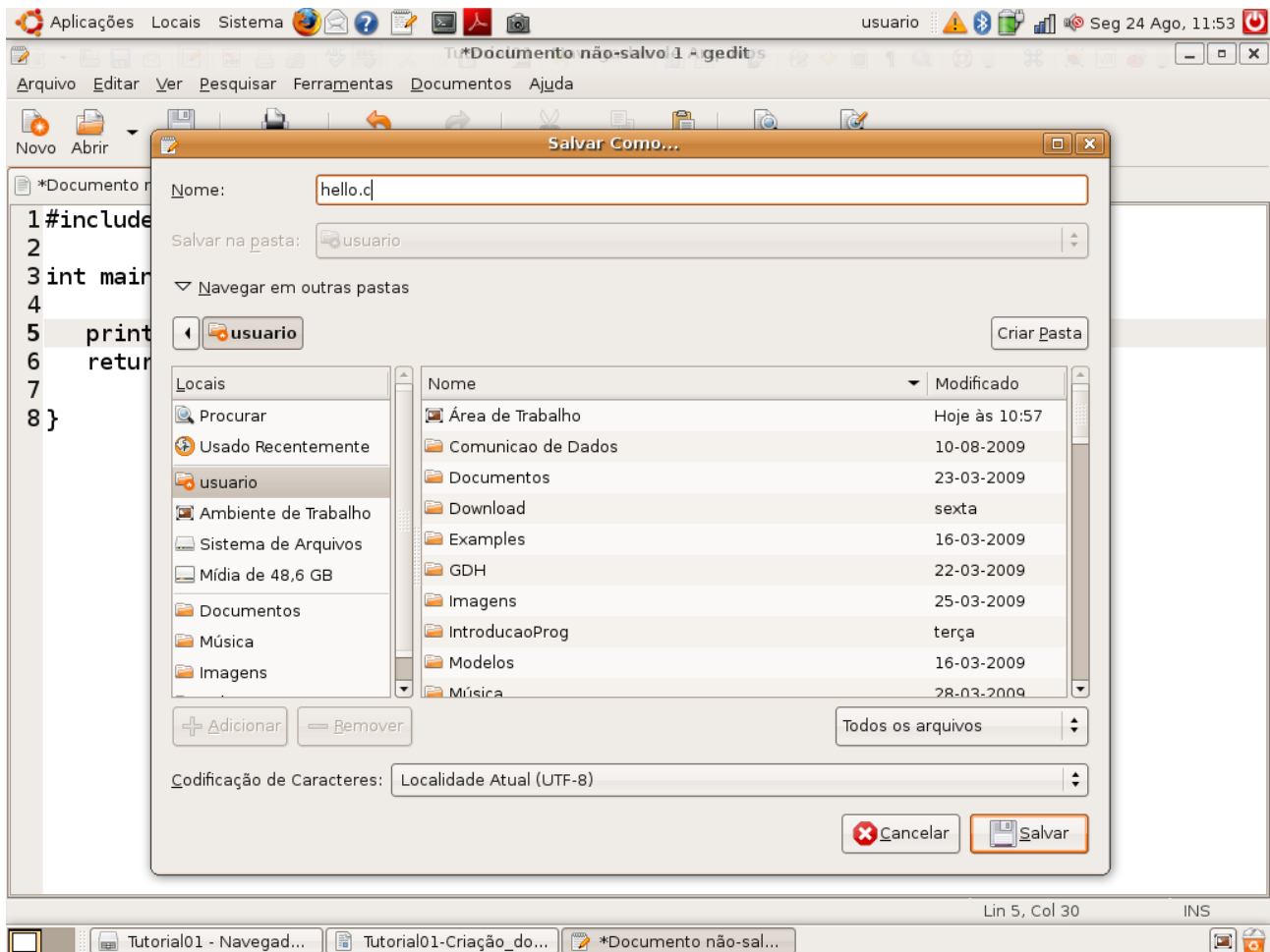
A screenshot of a gedit text editor window. The title bar shows the file path: "hello.c (~/IntroducaoProg/Linguagem C/EAD\_linguagemC/Exercicios/Tutorial01) - gedit". The menu bar includes "Arquivo", "Editar", "Ver", "Pesquisar", "Ferramentas", "Documentos", and "Ajuda". The toolbar contains icons for "Novo", "Abrir", "Salvar", "Imprimir...", "Desfazer", "Refazer", "Recortar", "Copiar", "Colar", "Localizar", and "Substituir". The main text area shows the following C code:

```
1#include <stdio.h>
2
3int main() {
4
5    printf("Hello, World!\n");
6    return 0;
7
8}
```

The status bar at the bottom indicates "Lin 5, Col 30" and "INS". The taskbar at the very bottom shows several open windows: "Tutorial01 - Navegad...", "Tutorial01-Criação\_do...", and "hello.c (~/Introducao...".

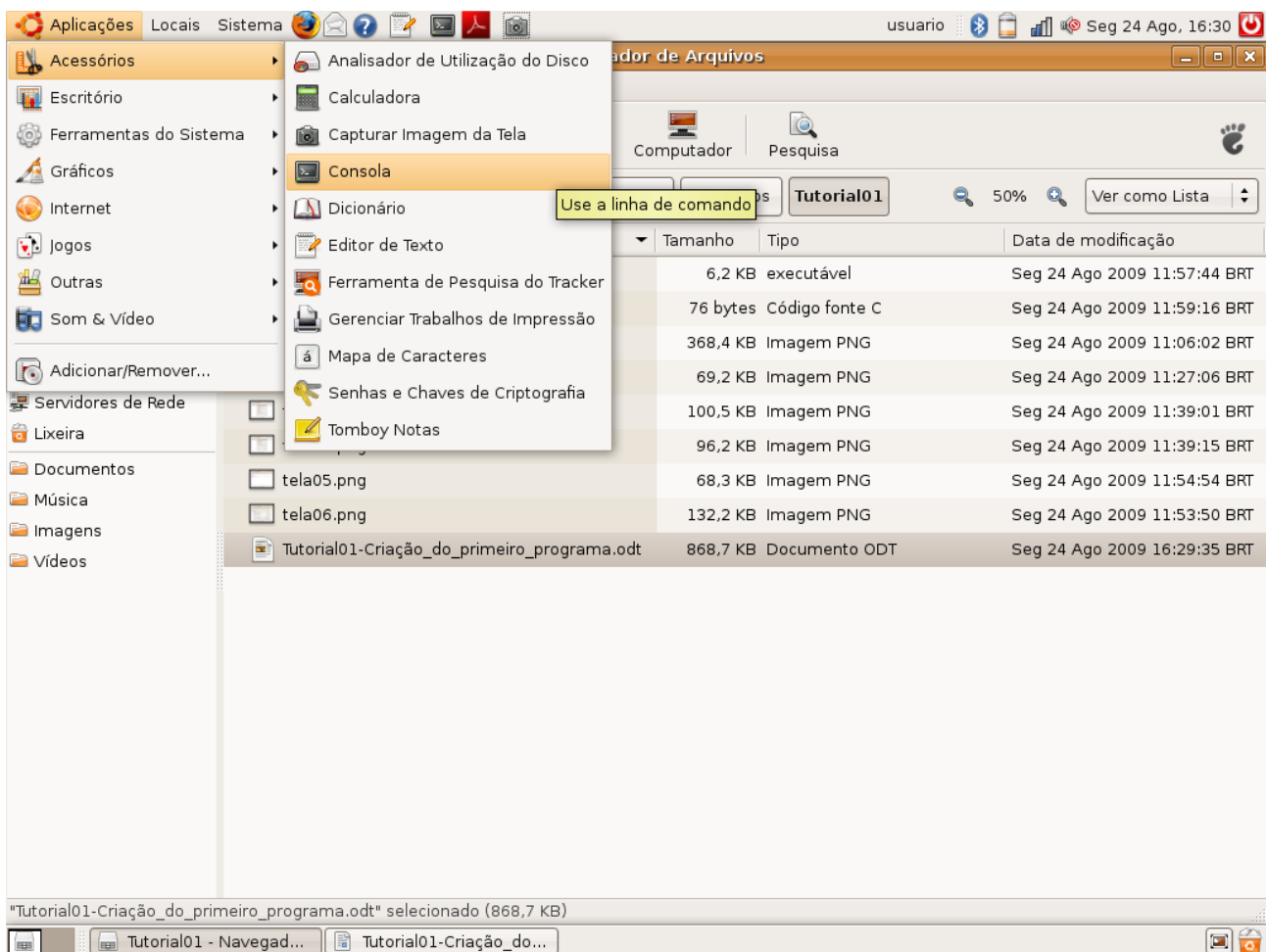
**Edição de programa fonte em linguagem C**

**Passo 4:** Salve o programa fonte com o nome de **hello.c**. Lembre que todo programa fonte na linguagem de programação C deve ter extensão **.c**.



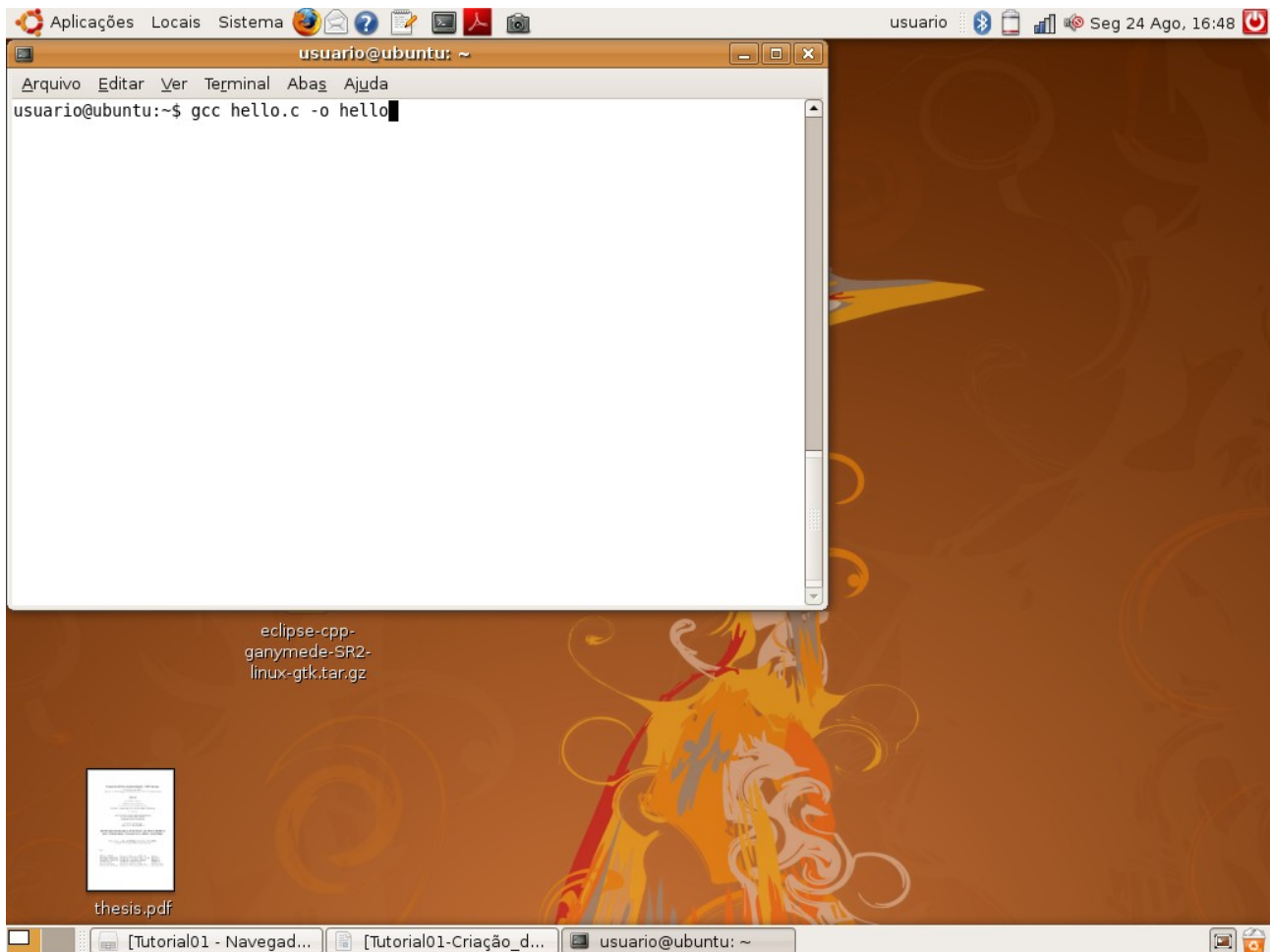
**Salvando o arquivo fonte.**

**Passo 5:** Abra uma tela de console(a) do Ubuntu. Vá em *Aplicações/Acessórios/Consola*. Veja a figura abaixo.



**Abertura de uma tela de console.**

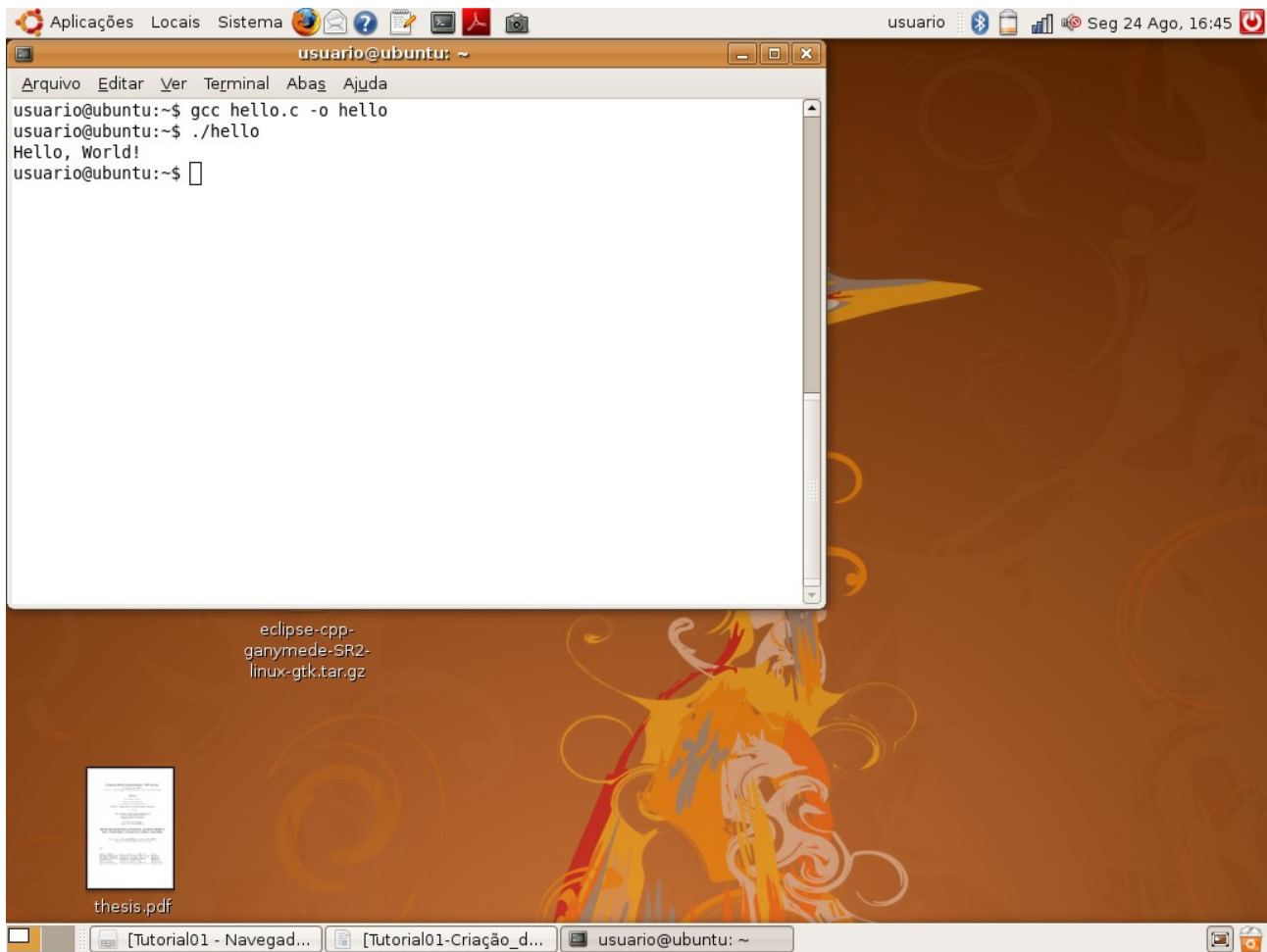
**Passo 6:** Compile o programa fonte. Digite na linha de comando: **gcc hello.c -o hello**, em seguida tecla <enter>. Neste passo você está chamando o compilador **gcc** e solicitando duas coisas: (1) que ele compile o programa fonte de nome **hello.c**, que deve estar no diretório corrente; (2) e que gere no diretório atual um programa executável com o nome **hello**. É através da diretiva **-o** que você diz ao compilador o nome que deve ter o programa executável. Lembre que em sistemas GNU/Linux os executáveis não devem ter extensão, ao contrário de sistemas Windows/DOS em que os executáveis têm extensão **.exe**.



*Compilação do arquivo fonte hello.c, gerando um arquivo executável hello.*



**Passo 7:** Execute o programa executável. Digite na linha de comando: **./hello**. O resultado deve ser semelhante ao da figura abaixo.



**Execução do programa executável hello.**

## ALGUNS ERROS COMUNS INFORMADOS PELO COMPILADOR NO PROCESSO DE COMPILAÇÃO:

1. **gcc: Hello.c: Arquivo ou diretório inexistente.** Este erro pode se dar por salvar o arquivo com um nome e chamar o compilador gcc para compilar um arquivo com nome diferente. Por exemplo, você salva o arquivo com o nome *hello.c* e tenta compilar o arquivo *Hello.c*. Note que um arquivo está escrito com letra minúscula e o outro com letra maiúscula. Então, neste caso, só existe na realidade o arquivo *hello.c*, levando o compilador a emitir o erro citado. Lembre que sistemas GNU/Linux, assim como o compilador gcc, são *case sensitives* (sensíveis ao caso), ou seja, distinguem arquivos ou variáveis escritos com letras maiúsculas de arquivos ou variáveis escritos com letras minúsculas.
2. **hello.c: Na função 'main':**  
**hello.c:6: erro: expected ';' before 'return'.** Através desta mensagem o compilador informa que na função 'main', na linha 6, ocorreu um erro, sendo esperado (*expected*) um ';' antes (*before*) do comando 'return'. Remetendo-se ao exemplo do programa *hello.c*, poderia estar faltando o ';' após a função `printf()`.
3. **hello.c: Na função 'main':**  
**hello.c:6: erro: expected declaration or statement at end of input.** Através desta mensagem o compilador informa que na função 'main', na linha 6, ocorreu um erro, sendo esperada uma declaração no final da entrada. Remetendo-se ao exemplo do programa *hello.c*, poderia estar faltando o '}' que é a chave de fechamento do programa.
4. **/tmp/ccwqVEXK.o: In function 'main':**  
**hello.c:(.text+0x19): undefined reference to 'print'**  
**collect2: ld returned 1 exit status.** Através desta mensagem o compilador informa (undefined reference to 'print') que na função 'main' há uma referência indefinida para 'print'. Isto ocorre porque não há função `print`, mas sim `printf`.

**OBSERVAÇÃO:** Lembre sempre que o compilador faz uma análise criteriosa da sintaxe de seu programa, ou seja, ele verifica se todas as palavras do seu código estão de acordo com o padrão da linguagem. Podemos fazer uma analogia do compilador com um professor rigoroso de português analisando uma redação, em que qualquer erro que ele encontre, por menor que possa parecer, como uma falta de vírgula ou uma única falta de acentuação, ele lhe peça para refazer sua redação corrigindo os erros apontados. No caso, o compilador C analisa um código em linguagem de alto nível que é escrito usando palavras chaves em inglês.