|  |  |
| --- | --- |
| **Opção** | **Significado** |
| **-Wall** | Liga grande quantidade de *warnings* do GCCEx.1: declarei uma variável mas não a usei, dará um *warning*Ex.:2: declarei a função main com uma quantidade de parâmetros diferente de 0 ou de dois parâmetros (argc e argv) |
| **-Wextra****Equivalente a escrever** **-W****Só que o nome –Wextra é mais descritivo** | Liga *warnings* extras, além daqueles ligados por –WallEx.: declarei uma função com algum parâmetro mas não usei os parâmetros declarados no código da função |
| **-ansi****Equivalente a escrever****-std=<padrão>** | Desabilita extensões do GCC à linguagem que estejam fora do padrão ANSI implementado no compilador gcc.Por exemplo, a utilização da palavra reservada **asm** para a inclusão de código assembly diretamente no código C.  |
| **-std** **Exemplo:****-std=c89****-std=c99****Depende dos padrões suportados pela versão do seu compilador *gcc* instalado** | Permite especificar qual o padrão definido para o compilador analisar o seu código fonte.Ex.: for(int i = 5; i<10; i++);Se compilar com: gcc –o x prog.c –pedanticAcusará um erro, pois o gcc tentará compilar no padrão definido por ele (modo c89 e, futuramente, gnu c99).Agora, se indicar que deve compilar no modo c99:gcc –o x progr.c –pedantic –std=c99então, não apresentará erro, pois o padrão c99 (ISO/IEC 9899:1999) já suporta esse tipo de construção semelhantemente ao padrão que tinha acabado de ser definido para o c++ (pode ser testado usando: **-std=c++98**).**Observação**: o gcc ainda não tem um suporte completo ao padrão c99, mas o padrão c99 do gcc já permite a utilização da biblioteca para números complexos **complex.h**. Informações retiradas por meio da digitação de: * **info gcc**
* **opção “m”, menu “Standards”**
 |
| **-pedantic** | Emite *warnings* sobre uso de itens proibidos pelo padrão ANSIEx.1: declarar o seguinte código gera um *warning* char v[n]; scanf(“%d”, &n);Ex.: misturar declaração de variáveis no meio do código printf(“%d”, 5); int x; |
| **-pedantic-errors** | Igual a opção pedantic, exceto que são geradas mensagens de erros, ao invés de mensagens de *warnings* |
| **-O2** | Liga o terceiro nível de otimização de código do GCC |
| **-o** | Especifica qual arquivo será gerado como saída (executável) |
| **-E** | Parar após a execução do pré-processamento. Não compila o arquivo. |
| **-x** | Especificar a linguagem em que os arquivos fontes devem ser processados. Exemplos:-xc-xc++**Observação 1**: se não for especificado esse parâmetro, a extensão do arquivo indicará ao gcc como deve ser entendido o arquivo. Se for “.c”, será considerado um código fonte em C. Se for “.cpp”, será considerado um código fonte em C++. **Observação 2**: este parâmetro não tem efeito se for escrito após o último arquivo de código fonte. *Exemplo sem efeito*: gcc –o prog prog1.c prog2.c -xc++*O certo seria*: gcc –o prog prog1.c –xc++ prog2.cougcc –o prog –xc++ prog1.c prog2.cougcc –xc++ -o prog prog1.c prog2.c |
| **-x none** | É equivalente a não especificar o parâmetro “-x”. O arquivo contendo o código fonte será tratado conforme a sua extensão “.c”, “.cpp”. |
| **-H** | Imprime o nome dos arquivos de cabeçalho conforme eles vão sendo usados |