UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

BCC35A - Linguagens de Programação

Prof. Dr. Rodrigo Hübner

Aula 09: *Bindings*, *Wrappers* e *Inline codes*

Bindings de programas `C/C++` em `Python`

- Iniciamos com a ferramenta `SWIG` (`Simplified Wrapper and Interface Generator`)
  - Permite a integração de código `C/C++` com várias linguagens de programação, incluindo `Python`
  - Gera automaticamente os wrappers necessários para conectar o código `C/C++` ao `Python`

Utilizando o `SWIG`...

- Necessário ter instalado o `SWIG` e o `python3-dev` (que fornece a biblioteca `Python.h`):

```
$ sudo apt install swig --assume-yes
$ sudo apt install python3-dev --assume-yes
```

- Criamos um arquivo de interface `SWIG` especificando como deve gerar os *wrappers* para o código `C/C++`.

- Vamos criar um arquivo chamado `exemplo.c` com o seguinte conteúdo:

```c
#include <stdio.h>

unsigned long long factorial(unsigned int n) {
    if (n == 0)
        return 1;
    else
        return n * factorial(n - 1);
}
```

- Definimos o conteúdo do cabeçalho `exemplo.h`:

```
unsigned long long factorial(unsigned int n);
```

- Agora vamos criar um arquivo de interface que define como o `SWIG` deve gerar a interface `Python` para a função em `C`. O arquivo deve se chamar `exemplo.i`:

```
%module exemplo
%{
#include "exemplo.c"
%}

%include "exemplo.c"
```

- Vamos usar o SWIG para gerar o wrapper Python a partir do arquivo de interface:
  ```
  $ swig -python exemplo.i
  ```
- Isso irá gerar um arquivo `exemplo_wrap.c` que contém o código necessário para conectar `Python` e `C`
- Agora, vamos compilar o código C junto com o arquivo `exemplo_wrap.c` gerado pelo `SWIG`:
  ```
  gcc -c -fpic exemplo.c exemplo_wrap.c -I/usr/include/python3.11
  ```
- Após a compilação, vamos criar o link para nosso código `C` compilado:
  ```
  gcc -shared exemplo.o exemplo_wrap.o -o _exemplo.so
  ```

- Programa `teste.py`:

```python
from exemplo import factorial as c_fact
import sys
from time import time

sys.setrecursionlimit(500000)

def py_fact(n):
    if n == 0:
        return 1
    else:
        return n * py_fact(n - 1)

N = 150000

a = time()
c_fact(N)
print(f'Tempo em C: {time() - a} segundos')

a = time()
py_fact(N)
print(f'Tempo em Python: {time() - a} segundos')
```

Inline `assembly` em um código `C`

- Para realizarmos código *inline* `assembly` dentro de um código `C`, podemos utilizar a palavra-chave `asm` seguida por uma `string` contendo as instruções `assembly`.
- Para exemplificar, vamos fazer um código simples que realiza a soma entre o valor de duas variáveis `a` e `b`.

```c
#include <stdio.h>

int main() {
    int a = 10;
    int b = 20;
    int result;

    asm("addl %1, %2;"
        "movl %2, %0;"
        : "=r" (result)
        : "r" (a), "r" (b)
        );

    printf("O resultado é: %d\n", result);

    return 0;
}
```

Próxima aula

- Acompanhamento de trabalho