Universidade Federal do Piauí
Campus Senador Helvídio Nunes de Barros - Picos
Disciplina: Algoritmos e Programaçao II
Curso: Bacharelado em Sistemas de Informação
Professor: Glauber Dias Gonçalves

Lista de Atividades 06: passagem de parâmetros e o qualificador *const*

Qualificador const da Linguagem C

É uma palavra-chave que informa ao compilador de C que o valor de uma variável não pode ser modificado. Esse qualificador pode ser utilizado na declaração de variáveis globais, substituindo a diretiva "#DEFINE" e variáveis locais dentro do escopo de funções ou como parâmetro de funções.

Um uso típico do qualificador *const* é a aplicação do *princípio do menor privilégio* na modularização de códigos. Esse princípio permite maior segurança no desenvolvimento de códigos em equipes, pois protege parâmetros de funções que necessitam apenas serem lidos de uma, possível, modificação acidental. Por exemplo, a função abaixo:

```
void imprime_string( const char * );

void main(){
	char s[50];
	printf("Informe uma palavra: ");
	scanf("%s", s);
	imprime_string(s);
}


void imprime_string( const char *s ){
	int i=0;
	while( s[i] != '\0' ){
		printf( "%c\n", s[i] );
		//s[i] = 'x';   // Não é permitido
		i++;
	}
}
```

A função *imprime_string* recebe uma string *s* como parâmetro, mais especificamente um apontador para *s*, e a imprime com um caractere por linha. Note que o conteúdo de *s* está protegido de modificações com o qualificador const, ou seja, o desenvolvedor que for implementar a função consegue apenas acessar o conteúdo de s, mas não pode modificá-lo.

Para mais exemplos e detalhes sobre o qualificador const veja a Seção 7.5 do livro "Como Programar em C" Deitell 6ª. Edição.

Nessa atividade utilizaremos o qualificador const para aplicar o princípio do menor privilégio em funções para manipulação de strings. Para isso, seis protótipos de funções da biblioteca string.h serão utilizados como modelo. São elas: *strcpy* e *strncpy* para copiar strings, *strcat* e *strncat* para concatenar strings e *strcmp* e *strncmp* para comparar strings.

Primeiramente, veja o uso dessas funções nos códigos de ajuda compartilhados no Sigaa, pasta 05_ponteiros códigos ex07.c, ex08.c e ex09.c

A seguir, faça a biblioteca *mystring* que implementa essas seis funções abaixo, seguindo estritamente os protótipos da biblioteca *string.h* da linguagem, que utiliza o princípio de menor privilégio. Ao ver o protótipo dessas funções e tentar implementá-los, você entenderá na prática como esse princípio é aplicado nessas funções! Os referidos protótipos podem ser encontrados nos slides 16 e 18 da Aula Ponteiros – parte 2 ou no livro do Deitell (6ª. edição) Sessões 8.6 e 8.7, ou qualquer bom website de programação em C (exemplos, http://br-c.org/ ,

https://cplusplus.com/). A sua biblioteca *mystring* deve ser implementada de forma modularizada em dois arquivos:

- *mystring.h*: é o cabeçalho de biblioteca e deve conter apenas os protótipos das seis funções.
- *mystring.c*: é a implementação da biblioteca as declarações das seis funções.

Peça ajuda aos monitores para compilar e testar sua biblioteca nesse formato com dois arquivos! Teste sua biblioteca com os exemplos de códigos ex07.c, ex08.c e ex09.c, verificando se eles funcionam igualmente como nas funções *string.h*. Poste apenas os arquivos *myString.c* e *myString.h* no Sigaa.