# Plano de Ensino

## Dados da Turma

Código da disciplina: **MCTA028-15**
Nome da disciplina: **Programação Estruturada**
Créditos: **(T-P-I) 2-2-4**
Carga horária: **48h**
Aulas práticas: **Sim**
Câmpus: **Santo André**
Docente responsável: **Prof. Daniel M. Martin**

Código da turma**: NB5MCTA028-15**
Turma: **NB5**
Turno: **noturno**
Quadrimestre: **3º**
Ano: **2019**

Alocação da turma:
**QUA 19:00 -- 21:00 sala S-006-0**
**SEX 21:00 -- 23:00 lab 408-2**

Horário para atendimento de dúvidas:
**SEX 19:00 -- 21:00 sala 536-2**

## Dados da disciplina

Objetivos gerais: que o aluno adquira familiaridade com uma linguagem de programação estruturada de nível intermediário com gerenciamento explícito memória. Que o aluno aprofunde suas habilidades de modelagem e implementação de algoritmos para solucionar problemas computacionais.

Objetivos específicos: apresentar noções básicas e intermediárias sobre algoritmos, programação em linguagens compiladas, compilação, programas em execução (processos), ponteiros, alocação estática e dinâmica de memória, vetores e matrizes, funções e passagem de parâmetros, registros, arquivos e recursividade. Aplicar todos os conceitos apresentados no contexto da resolução de problemas clássicos e novos da computação.

Bibliografia:

1. KERNIGHAN, Brian W.; RITCHIE, Dennis M.. The C programming language. 2. ed. Englewood Cliffs, USA: Prentice Hall, c1988. xii, 272. (Prentice-Hall software series). ISBN 9780131103627.
2. SEDGEWICK, Robert. Algorithms in C, vol. 1-4: part 1-4: fundamentals, data structures, sorting, searching. 3. ed. Reading, USA: Addison-Wesley, c1998. xvii, 702. ISBN 9780201756081.
3. PINHEIRO, Francisco A. C. Elementos de programação em C. Porto Alegre, RS: Bookman, 2012. xx, 528 p., il. ISBN 9788540702028.
4. FEOFILOFF, Paulo. Algoritmos em linguagem C. Rio de Janeiro, RJ: Elsevier, 2009. 208 p. ISBN 9788535232493.
5. TANENBAUM, A. M.; LANGSAM, Y.; AUGENSTEIN, M. J. Estruturas de dados usando C. São Paulo, SP: Pearson Makron Books, 1995.

# Programa

**QUA 25/09** -- *teórica* -- Apresentação da disciplina; o que é computação; problemas computacionais; problema x instância; linguagem de programação: baixo nível x alto nível; linguagem de máquina; linguagem assembly; linguagem estruturada x não estruturada; linguagem imperativa x declarativa; linguagem C: história, uso, pontos positivos e negativos. Comparação C x Java.

**SEX 27/09** -- *prática* -- Detalhes da linguagem C, tipos de dado, tamanho ocupado na memória por cada tipo, declaração de variável, inicialização de variável, declaração de funções, declaração de vetor, inicialização de vetor. Saída de dados (printf); texto-formato. Entrada de dados (scanf); texto-formato. Sintaxe comum de C e Java: for, while, switch, if … else, break, continue. Operadores aritméticos; operadores lógicos; operadores bit-a-bit.

**QUA 02/10** -- *teórica* -- Funções e procedimentos; o tipo void; chamada de função; passagem de parâmetros (sempre por valor); escopo de variáveis (local, global); ambiente de execução (stack, heap); chamada de função; stack frame; representação de vetores na memória; passagem de vetor como parâmetro; impressão de endereço de memória. Representação de matrizes na memória; passagem de matriz como parâmetro de função.

**SEX 04/10** -- *prática* -- Redirecionamento de entrada e saída; Mais problemas de implementação do URI (vetores e matrizes). O problema das frases palíndromas.

**QUA 09/10** -- *teórica* -- Recursão

**SEX 11/10** -- *prática* -- Recursão

**QUA 16/10** -- *teórica* -- Ponteiros e alocação dinâmica

**SEX 18/10** -- *prática* -- Ponteiros e alocação dinâmica

QUA 23/10 -- *teórica* -- P1

SEX 25/10 -- *prática* -- Ponteiros, alocação dinâmica

QUA 30/10 -- *teórica* -- Registros (structs)

SEX 01/11 -- *prática* -- Aplicação de registros para resolução de problemas de geometria computacional

QUA 06/11 -- *teórica* -- Listas simplesmente ligadas

SEX 08/11 -- *prática* -- Listas ligadas I

QUA 13/11 -- *teórica* -- Listas duplamente ligadas. Alocação progressiva de memória

SEX 22/11 -- *prática* -- Listas ligadas II

QUA 27/11 -- *teórica* -- Manipulação de arquivos

SEX 29/11 -- *prática* -- Manipulação de arquivos

QUA 04/12 -- *teórica* -- Projeto de algoritmos (aplicação dos conceitos do curso)

SEX 06/12 -- *prática* -- Projeto de algoritmos (aplicação dos conceitos do curso)

QUA 11/12 -- *teórica* -- P2

SEX 13/12 -- *prática* -- SUB

QUA 18/12 -- *prática* -- REC

SEX 20/12 -- *teórica* -- Vista de prova

# Avaliações

Haverá duas provas em sala de aula (da parte teórica), a primeira (**P1**) no dia 23/10 e a segunda (**P2**) no dia 11/12. Cada prova compreende toda a matéria dada até sua data. Além disso, as atividades da aula prática de cada semana contarão nota (com diferentes, de acordo com a dificuldade); a média ponderada de todas as atividades práticas do quadrimestre será denotada por **PR**.

A média final do curso (pré-recuperação) será calculada por

**N** = 20% **P1** + 30% **P2** + 50% **PR**

O aluno que exceder 6 faltas terá seu conceito registrado como O.

Para alunos com pelo menos 75% de frequência (no máximo 12h de faltas) os conceitos serão atribuídos de acordo com a tabela abaixo.

| MÉDIA FINAL (N) | CONCEITO |
|---|---|
| 8.5 ≤ N | A |
| 7.0 ≤ N < 8.5 | B |
| 5.5 ≤ N < 7.0 | C |
| 4.5 ≤ N < 5.5 | D |
| N < 4.5 | F |

Haverá também uma prova SUB e uma prova REC, 100% práticas, que compreendem toda a matéria.

### SUB 13/12

Uma prova que você poderá fazer se, justificadamente, perder pelo menos uma das provas anteriores. As regras (lista de justificativas aceitáveis) para se poder fazer a SUB estão definidas na Resolução ConsEPE 227. O peso da SUB na média final será de acordo com o(s) peso(s) da(s) prova(s) que ela estiver substituindo. O conteúdo da SUB é a matéria toda.

### REC 18/12

Uma prova, aberta a todos os alunos que desejarem melhorar o conceito final. Esta avaliação seguirá as regras definidas na Resolução ConsEPE 182. Se você não for reprovado por faltas, e tiver obtido um conceito diferente de A, poderá fazer a REC para tentar melhorar a média final, mas se entregar a REC, então ela necessariamente entrará para o compto da média e conceito finais, seja isso para melhor ou para pior. O conteúdo da REC é a matéria toda.

Se a média final antes da REC é **N**, então a nova média final (i.e. depois da REC) será calculada por (**N** + **REC**) / 2.

## Correção das atividades práticas

Em cada aula o professor irá designar atividades de implementação que devem ser realizadas e entregues até a data limite estabelecida, normalmente até antes da aula prática da semana seguinte.

Cada atividade prática deverá ser submetida pelo site da disciplina na plataforma Moodle (instalado e mantido pelo CMCC). Nesse site, cada atividade conta com um corretor automático que compila, executa e verifica a corretuda do programa submetido na mesma hora e atribui uma nota. Se a nota for baixa, o aluno tem a oportunidade de examinar seu código, depurá-lo e submeter nova versão, quantas vezes achar necessário, até a data limite da atividade.

Por padrão, essas atividades devem ser feitas individualmente. O sistema de submissão de atividades conta com um software detector de plágio que funciona muito bem! Portanto, o aluno que entregar um programa plagiado da internet ou de um colega -- ainda que ele tenha feito alterações na ordem e nos identificadores do programa -- obterá nota zero na avaliação correspondente. O aluno que deixou seu código ser plagiado ou que "passou" seu código ao colega também receberá nota zero. Portanto, não divulgue seu seu código! Proteja-o!

Além das atividades práticas semanais, o professor pode pedir projetos maiores de implementação, chamados de exercícios-programas ou EPs, que costumam ter prazos também maiores. Outra alternativa é a utilização de sites de correção automática como o URI, SPOJ, UVA e outros.

O peso de cada avaliação prática será definido de acordo com seu grau de dificuldade.