Departamento de Sistemas de Computação
Universidade de São Paulo
SSC 502 – Laboratório de Introdução a Ciência de Computação I

Aula 1
Apresentação do Curso

Responsável

Prof. Seiji Isotani (sisotani@icmc.usp.br)

Veterano Voluntário

Bruno Orlandi (brorlandi@grad.icmc.usp.br)

Informações sobre o curso

- Prof. Seiji Isotani - sala 3-157 – sisotani@icmc.usp.br

- Site do curso http://agora.tidia-ae.usp.br/
  - Para quem nunca acessou: username e login = n* USP
- Material
  - Todo o material apresentado em aula será disponibilizado neste site

Objetivos do Curso

- Praticar os conceitos vistos na disciplina de ICC
- Programação em linguagem estruturada
  - Linguagem C
- Ambientes de programação
  - codeblocks
- Depuração

ATENÇÃO

- Para quem já sabia linguagem de programação
  - Não adianta apenas saber programar
  - É preciso criar algoritmos
- Para quem não sabia programar
  - Ajuda a pensar e desenvolver habilidades para
  - criar algoritmos

Conteúdo do curso

- **Resolução de problemas** e **desenvolvimento de programas**: análise e solução de problemas, representação e documentação
- **Estruturas de programas**: decisão e repetição
- **Tipos de dados simples**
- **Modularização de programas**: procedimentos, funções e passagem de parâmetros
- **Tipos de dados compostos**: vetores, matrizes, cadeias de caracteres, registros, conjuntos e estruturas dinâmicas (ponteiros)
- **Arquivos**

Metodologia

- Resolução de exercícios em laboratório
- Atividades em Duplas
  - Duplas diferentes a cada aula
- Entrega dos Exercícios EM AULA via tidia
  - Ambos os alunos precisam postar o exercício no Tidia
- Projeto da disciplina

Ao final do curso espera-se que ....

- Seja capaz de analisar um problema e transformá-lo em algoritmo
- Seja capaz de traduzir um algoritmo para a linguagem C
- Seja capaz de trabalhar em grupo
- Seja criativo
- Seja autodidata
- Saiba **MAIS** que o professor

Avaliação

- Exercícios em sala de aula (Ex)
- 2 Provas (P1, P2)
- Trabalho em grupo: (T)
- Cálculo da Média Final (M)

$$(Ex + P1 + P2 + T)/4$$

Recuperação

- Se Exer **OU** Trab ou Prova < 5 **RECUPERAÇÃO**
- Se M ≥ 5 e frequência ≥ 70% : **APROVADO**
- Se 3 ≤ M < 5 e frequência ≥ 70% **RECUPERAÇÃO**
- Se M < 3 **ou** frequência < 70% **REPROVADO**

Presença e Prova Substitutiva

- O controle de presença é um requisito imposto pela USP, assim o controle será feito por meio de listas
- NÃO haverá prova substitutiva (sub)
  - Caso falte uma aula, é preciso conversar com o professor para pedir um exercício Extra para entrega via Tidia.

Mão a Obra

- Tente Acessar o Tidia
  - http://agora.tidia-ae.usp.br/
- Ambiente de Programção
  - Codeblocks (Windows) - Principal
  - Editor de texto (e.g. emacs – Linux)

Cronograma das aulas

- Será disponibilizado sexta-feira via Tidia

Meu Primeiro Programa

Seu primeiro programa

```
#include < stdio.h>

int main() {
        printf("\nHello World\n");
        return 0;
}
```

1. Logar no Windows (é o padrão que vamos usar)
2. Entrar no CodeBlocks
3. Editar e salvar o programa hello.c
4. Compilar/executar

1. Logar no Linux
2. Editar e salvar o programa hello.c
3. Compilar/executar

```
gcc hello.c -o hello
./hello
```

Aviso

- Não se esqueça de salvar os projetos e exercícios no final da aula em uma pasta privada (não no ambiente compartilhado como desktop) pois não há garantia alguma que na próxima aula seus dados estarão ainda aqui!

Linguagem de Programação C

Sumário

- Estrutura de programas (sequenciais)
- Tipos de dados simples
- Declaração de variáveis
- Entrada/Saída (E/S)
- Operadores e funções pré-definidas
- Exercícios

Linguagem C

- Linguagem de propósito geral conhecida por ser eficiente, econômica e portável
- Não é uma linguagem amigável para se aprender programação

Linguagem C

- Um programa em C é uma coleção de diretivas, declarações, definições, blocos de comandos/instruções e funções
- Um programa pode ser dividido em um ou mais arquivos fontes
  - É necessário compilar cada arquivo fonte e fazer o “link” dos arquivos objetos resultantes para tornar um programa executável
  - Constantes e macros são normalmente organizadas em arquivos separados conhecidos como “header files”ou “include files

Estrutura do Programa

```
#include <nome_da_biblioteca>
int main() {
        instrução 1;
        instrução 2;
        instrução n;
}
```

- As instruções são executadas sequencialmente a partir da função main(), até que uma instrução de desvio ou de retorno seja encontrada

Declaração de variáveis

- O que são variáveis?
  - São referências a áreas de memória do computador que armazenam dados de interesse do programador, dados esses que podem ser alterados a qualquer momento
- A declaração de variáveis é definida pelo programador de acordo com a necessidade para a resolução do problema

Declaração de variáveis

- Cada variável possui um tipo. Exemplo:
  - Inteiros:
    - int x; int x, y;
  - Números Reais
    - float z;
  - Caracteres
    - char inicial; char string[100];
- A linguagem C possui cinco tipos básicos:
  - int, float, double, void, char

Declaração de variáveis

- Podem ter até 32 caracteres
- Devem começar com letra ou sublinhado (_), sendo que os caracteres subsequentes devem ser letras, números ou sublinhado (_)
- Não podem coincidir com nomes de palavras reservadas, nem de funções declaradas pelo programador ou em bibliotecas do C
- C é "case sensitive“, ou seja, maiúsculas são distintas de minúsculas (Nome ≠ nome ≠NOME ≠ NoMe)

Comandos de Atribuição

- Para atribuir um conteúdo a uma variável, utiliza-se a seguinte instrução:
  - <variável> = <conteúdo>
- O conteúdo atribuído à variável pode ser uma literal (número ou caracter) ou outra variável, e seu tipo deve ser compatível com o tipo da variável
- Algumas variações nesse formato são admitidas Por exemplo:
  - <variável1> = <variável2> = <conteúdo>;

Comandos de Atribuição

- Exemplo de atribuições de valores ou operações a variáveis (sinal de igualdade)
  - x = 4; x = x + 2; y = 2.5; sexo = ‘F’;
- • Em C um caractere é representado entre apóstrofos e uma cadeia de caracteres entre aspas
- Para armazenar uma cadeia de caracteres numa variável deve-se utilizar uma função para manipulação dos mesmos.
  - strcpy(nome, “Joao”)

Exercício

- Crie um programa que possui 3 variáveis do tipo inteiro.
  - A primeira variável vale 2
  - A segunda vale 10
  - E a Terceira é a multiplicação das variáveis anteriores.
  - Imprima a terceira variável utilizando o comando:
  - printf (“%d \n”, *<nome da variável>*);

Comandos de entrada e saída básicos

- Comando de entrada recebe dados digitados pelo usuário e de saída mostra os dados na tela
- Comandos de entrada mais utilizados:
  - gets(nome); => le uma cadeia de caracteres
  - scanf(“%d”, &x);
- Comando de saída
  - printf (“%d”, x);
- **%d** para int; **%f** para float; **%c** para caracter; **%s** para conjunto de caracteres (string).

Exemplo

```
#include <stdio.h>
int main () {
        int i;
        printf (“Digite um número: ");
        scanf(“%d”, &i);
        printf ("\n\n o número lido foi %d“, i);
        return 0;
}
```

- Agora tente ler e imprimir uma variável do tipo float

Exercício

- Crie um programa que lê o valor de duas variáveis do tipo inteiro e imprime a multiplicação e a divisão deles.

Referências

- Ascencio AFG, Campos EAV. Fundamentos de programação de computadores. São Paulo : Pearson Prentice Hall, 2006. 385 p.
- Material do Prof. Dr. Jó Ueyama
  - http://wiki.icmc.usp.br/index.php/SSC-102
- Outras fontes interessantes
  - http://mtm.ufsc.br/~azeredo/cursoC/c.html
  - http://www.cs.cf.ac.uk/Dave/C/
  - http://msdn.microsoft.com/en-us/library/aa315845(VS.60).aspx
  - http://www.acm.uiuc.edu/webmonkeys/book/c_guide/
  - http://techpubs.sgi.com/library/manuals/0000/007-0701-150/pdf/007-0701-150.pdf