MC-102 — Aula 08
Vetores e Strings

Instituto de Computação – Unicamp

23 de Março de 2012

Roteiro

Vetores

Como armazenar 3 notas?

```
float nota1, nota2, nota3;

printf("Nota do aluno 1: ");
scanf("%f", &nota1);
printf("Nota do aluno 2: ");
scanf("%f", &nota2);
printf("Nota do aluno 3: ");
scanf("%f", &nota3);
```

Vetores

Como armazenar 100 notas?

```
float nota1, nota2, nota3, /* .... */ nota100;

printf("Nota do aluno 1: ");
scanf("%f", &nota1);
printf("Nota do aluno 2: ");
scanf("%f", &nota2);

/* ... */

printf("Nota do aluno 100: ");
scanf("%f", &nota100);
```

Vetores — Definição

Coleção de variáveis do mesmo tipo referenciada por um nome comum.

(Herbert Schildt)

- acesso por meio de índice inteiro
- posições contíguas na memória
- tamanho pré-definido
- índices fora dos limites podem causar comportamento anômalo do programa.

Declaração de um vetor

<tipo> identificador [<tamanho do vetor>];

Exemplo

```
float notas[100];
int medias[100];
char nome[200];
```

Usando um vetor

Após declarada uma variável do tipo vetor, pode-se acessar uma determinada posição do vetor utilizando um valor inteiro.

identificador [<posição>];

- O acesso de um vetor em uma posição específica tem o mesmo comportamento que uma variável simples.
- A primeira posição de um vetor tem índice 0.

Usando um vetor

- A última posição de um vetor tem índice *<tamanho do vetor>* - 1.

Exemplo

```
int nota[10];
int a;
nota[5] = 95;
a = nota[5];
```

Usando um vetor

identificador [<posição>];

- Você pode usar valores inteiros para acessar uma posição do vetor.
- O valor pode ser inclusive uma variável inteira.

Exemplo

```
int g, vet[10];
for(g=0; g<10; g++)
    vet[g]=5*g;
```

Vetores

- Na memória:

```
int d;
int vetor[5];
int f;
```

Nome	d	vetor					f
Índice	-	0	1	2	3	4	-

Vetores

- Ao executar `vetor[3]=10;`:

Nome	d	vetor					f
Índice	-	0	1	2	3	4	-
					10		

Vetores

- O que ocorre se for executado os comandos:

```
vetor[5]=5;
vetor[-1]=1;
```

Vetores

- Ao executar

```
vetor[3]=10;
vetor[5]=5;
vetor[-1]=1;
```

Nome	d	vetor					f
Índice	-	0	1	2	3	4	-
	1				10		5

- Isto irá causar um erro no seu programa pois você está alterando valores de outras variáveis.
- Em muitos casos o seu programa irá ser encerrado (Segmentation Fault).

Questões importantes sobre vetores

- O tamanho do vetor é pré-definido (durante a execução do programa não pode ser alterado).
- Índices fora dos limites podem causar comportamento anômalo do código.

Como armazenar n (<= 100) notas?

```
float nota[100];
int n, i;

printf("Número de alunos: ");
scanf("%d", &n);

for (i = 0; i < n; i++) {
  printf("Nota do aluno %d: ", i+1);
  scanf("%f", &nota[i]);
}
```

- O programa acima está correto?

Como armazenar n (<= 100) notas?

- Você deve testar se $n > 100$ para evitar erros!!

```
float nota[100];
int n, i;
printf("Número de alunos: ");
scanf("%d", &n);
if(n>100){
  n=100;
  printf("\nNumero maximo de alunos alterado para 100");
}
for (i = 0; i < n; i++) {
  printf("Nota do aluno %d: ", i+1);
  scanf("%f", &nota[i]);
}
```

Produto Interno de dois vetores

- Ler dois vetores de dimensão 5 e computar o produto interno destes.
- Quais tipos de variáveis usar?

Produto Interno de dois vetores

Ler dois vetores de dimensão 5 e computar o produto interno destes.

```
int main(){
  double vetor1[5], vetor2[5], resultado;
  int i;
  for(i=0; i<5; i++){
    printf("Entre com valor %d para vetor 1:",i+1);
    scanf("%lf",&vetor1[i]);
  }
  for(i=0; i<5; i++){
    printf("Entre com valor %d para vetor 2:",i+1);
    scanf("%lf",&vetor2[i]);
  }
  //calculando o produto interno
  resultado = 0.0;
  for(i=0; i < 5; i++){
    resultado = resultado + ( vetor1[i]*vetor2[i] );
  }
  printf("\n\nO produto interno e: %lf\n",resultado);
}
```

Elementos Iguais

- Ler dois vetores com 5 inteiros cada.
- Checar quais elementos do segundo vetor são iguais a algum elemento do primeiro vetor.

Elementos Iguais

```
int main(){
  int vetor1[5], vetor2[5];
  int i, j,  umEmComum;
  for(i=0; i<5; i++){
    printf("Entre com valor %d para vetor 1:",i+1);
    scanf("%d",&vetor1[i]);
  }
  printf("\n\n");
  for(i=0; i<5; i++){
    printf("Entre com valor %d para vetor 2:",i+1);
    scanf("%d",&vetor2[i]);
  }
  ...
```

Elementos Iguais

```
  ...
  umEmComum = 0;
  for(i = 0; i < 5 ; i++)
    for(j = 0; j < 5; j++)
      if(vetor1[i] == vetor2[j]){
         umEmComum = 1;
         printf("Posicao %d do vetor1 igual a %d do vetor2.\n",i,j);
      }
  if(!umEmComum)
    printf("Nenhum elemento em comum!\n");
}
```

Strings

- A linguagem C não possui o tipo *string* explicitamente mas podemos considerar um vetor de caracteres como uma *string*.
- Em C uma string é sempre terminada pelo caracter especial: `'\0'`
- **Portanto sempre declare uma string com um caracter a mais do que precisa!**.
- Se por exemplo estivermos trabalhando com strings de 10 caracteres:

```
char st[11];
```

Strings

- Para ler ou imprimir uma string do teclado usamos o operador especial $\%s$.

```
int main(){
   char st[80];
   int a;

   printf("\nEntre com nome:");
   scanf("%s",st);
   printf("\nEntre com idade:");
   scanf("%d",&a);
   printf("\n Digitado: %s e %d\n",st,a);
}
```

- Note que para strings não é utilizado o & no comando *scanf*.

Exemplos Strings

- Ler uma string de até 80 caracteres e salvar a inversa desta em um vetor.
- Imprimir a inversa da string lida.

Exemplos Strings

```
int main(){
  char st1[80], stInversa[80];
  int i, j , tam;
  printf("Digite um texto (max. 80):");
  scanf("%s",st1);
  tam=0;
  while(st1[tam] != '\0' && tam < 80){
    tam++;
  }
  stInversa[tam] = '\0';
  for(j = tam-1, i = 0 ; j >= 0 ;  j--, i++){
    stInversa[j] = st1[i];
  }
  printf("A inversa:%s\n", stInversa);
}
```

Exemplos Strings

- Ler uma string de até 80 caracteres e salvar a inversa desta em um vetor.
- Imprimir a inversa da string lida.
- **Não usar vetor adicional!**

Exemplos Strings

```
int main(){
  char st1[80], aux;
  int i, j, tam;
  printf("Digite um texto (max. 80):");
  scanf("%s",st1);
  tam=0;
  while(st1[tam] != '\0' && tam < 80){
    tam++;
  }
  i = 0;
  j = tam -1;
  while(i < j){
    aux = st1[i];
    st1[i] = st1[j];
    st1[j] = aux;
    i++; j--;
  }
  printf("A inversa:%s\n",st1);
}
```

fgets

- Ao usar o comando *scanf* para ler uma string, você deve garantir que foi alocado uma string de tamanho suficiente para armazenar todos os caracteres.
- Caso o usuário digite mais caracteres do que o tamanho alocado, isto poderá ocasionar erros!
- Uma alternativa para ler strings é usar o comando *fgets()*.

```
fgets( <identificador>, TAM, stdin);
```

onde <*identificador*> é a variável, TAM é um inteiro indicando até quantos caracteres devem ser lidos (até TAM-1 são lidos e um é reservado para o barra zero).

Exemplos fgets

```
int main(){
  char st1[80];

  printf("Digite um texto (max. 80):");
  fgets(st1,80,stdin);

  printf("Digitado: %s",st1);
}
```

Problemas fgets

- No exemplo abaixo, não será lido a string como deveria.
- Por que?

```
int main(){
  char st1[80];
  int n;
  printf("Digite um numero:");
  scanf("%d",&n);
  printf("Numero digitado: %d\n",n);
  printf("Digite um texto (max. 80):");
  fgets(st1,80,stdin);
  printf("Digitado: %s",st1);
}
```

Problemas fgets

- O comando **scanf** não lê o "enter" final quando o usuário digita um número.
- Este "enter" fica armazenado no *buffer* do teclado.
- O comando **fgets** então lê o "enter" do buffer, e entende que foi isso a string digitada pelo usuário.
- Como resolver este problema?

Problemas fgets

- Uma maneira de resolver este problema é sempre limpar o *buffer* do teclado antes de usar o comando **fgets**.
- Use um dos seguintes comandos:
  - fflush(stdin);
  - setbuf(stdin, 0);

Problemas fgets

```
int main(){
  char st1[80];
  int n;

  printf("Digite um numero:");
  scanf("%d",&n);
  printf("Numero digitado: %d\n",n);

  printf("Digite um texto (max. 80):");
  setbuf(stdin, 0);
  fgets(st1,80,stdin);

  printf("Digitado: %s",st1);
}
```

Exercício

- Escreva um programa que lê uma string de até 50 caracteres, e imprime "Palindromo" caso a string seja um palindromo e "Nao Palindromo" caso contrário.
- OBS: Um palindromo é uma palavra ou frase, que é igual quando lida da esquerda para a direita ou da direita para a esquerda (acentos e espaços em brancos são descartados).
- Exemplo de palindromo: Saudável leva duas.