Unidade I – Uma Introdução Através de Exemplos

Disciplina Linguagens de Programação I

Bacharelado em Ciência da Computação da Uerj

Professores Guilherme Mota e Leandro Marzulo

ANSI C

```
#include  <stdio.h>
int main()
{
   printf("Hello World!\n");
}
```

Primeiro Programa

Primeiro Programa

nome.c

```c
#include <stdio.h>
int main ()
{
   printf("Hello World!\n");
}
```

```
> gcc  nome.c
> ./a.out
```

Primeiro Problema

Primeiro Problema

- Escreva um programa que imprima uma tabela dos valores de temperatura em Fahrenheit e Celsius.
  - Valor inicial: 0 °F
  - Valor Final: 300 °F
  - Incremento: 20 °F

$$C = \frac{5}{9} \times (F - 32)$$

Fahrenheit	Celsius
0	-17.8
20	-6.7
40	4.4
60	15.6
...	...
300	148.9

Primeiro Problema

```c
#include <stdio.h>
int main() /* print Fahrenheit-Celsius table */
{
  printf("%3d %6.1f\n", 0, (5.0 / 9.0)*(0 - 32));
  printf("%3d %6.1f\n", 20, (5.0 / 9.0)*(20- 32));
  printf("%3d %6.1f\n", 40, (5.0 / 9.0)*( 40-32));
  printf("%3d %6.1f\n", 60, (5.0 / 9.0)*(60-32));
  printf("%3d %6.1f\n", 80, (5.0 / 9.0)*(80-32));
  printf("%3d %6.1f\n", 100, (5.0 / 9.0)*(100-32));
  printf("%3d %6.1f\n", 120, (5.0 / 9.0)*(120-32));
  printf("%3d %6.1f\n", 140, (5.0 / 9.0)*(140-32));
  printf("%3d %6.1f\n", 160, (5.0 / 9.0)*(160-32));
  printf("%3d %6.1f\n", 180, (5.0 / 9.0)*(180-32));
  printf("%3d %6.1f\n", 200, (5.0 / 9.0)*(200-32));
  printf("%3d %6.1f\n", 220, (5.0 / 9.0)*(220-32));
  printf("%3d %6.1f\n", 240, (5.0 / 9.0)*(240-32));
  printf("%3d %6.1f\n", 260, (5.0 / 9.0)*(260-32));
  printf("%3d %6.1f\n", 280, (5.0 / 9.0)*(280-32));
  printf("%3d %6.1f\n", 300, (5.0 / 9.0)*(300-32));
}
```

```
$ ./progPag026
  0  -17.8
 20   -6.7
 40    4.4
 60   15.6
 80   26.7
100   37.8
120   48.9
140   60.0
160   71.1
180   82.2
200   93.3
220  104.4
240  115.6
260  126.7
280  137.8
300  148.9
$
```

Críticas à Solução

- **Trabalhosa**
- **Longa**
- **Repetitiva**
- **Pouca legibilidade**
- **Difícil manutenção**
- **Não reaproveitável**

```
#include  <stdio.h>
int main() /* print Fahrenheit-Celsius table */
{
   printf("%3d %6.1f\n", 0, (5.0 / 9.0)*(0 - 32));
   printf("%3d %6.1f\n", 20, (5.0 / 9.0)*(20- 32));
   printf("%3d %6.1f\n", 40, (5.0 / 9.0)*( 40-32));
   printf("%3d %6.1f\n", 60, (5.0 / 9.0)*(60-32));
   printf("%3d %6.1f\n", 80, (5.0 / 9.0)*(80-32));
   printf("%3d %6.1f\n", 100, (5.0 / 9.0)*(100-32));
   printf("%3d %6.1f\n", 120, (5.0 / 9.0)*(120-32));
   printf("%3d %6.1f\n", 140, (5.0 / 9.0)*(140-32));
   printf("%3d %6.1f\n", 160, (5.0 / 9.0)*(160-32));
   printf("%3d %6.1f\n", 180, (5.0 / 9.0)*(180-32));
   printf("%3d %6.1f\n", 200, (5.0 / 9.0)*(200-32));
   printf("%3d %6.1f\n", 220, (5.0 / 9.0)*(220-32));
   printf("%3d %6.1f\n", 240, (5.0 / 9.0)*(240-32));
   printf("%3d %6.1f\n", 260, (5.0 / 9.0)*(260-32));
   printf("%3d %6.1f\n", 280, (5.0 / 9.0)*(280-32));
   printf("%3d %6.1f\n", 300, (5.0 / 9.0)*(300-32));
}
```

Variáveis e aritmética

Variáveis e Aritmética

```
/* progPag021.c */
#include <stdio.h>
/* print Fahrenheit-Celsius table for fahr = 0, 20, ..., 300 */
int main()
{
   int fahr, celsius;
   int lower, upper, step;
   lower = 0;   /* lower limit of temperature scale */
   upper = 300; /* upper limit */
   step = 20;   /* step size */

   fahr = lower;
   while (fahr <= upper)
   {
      celsius = 5 * (fahr-32) / 9;
      printf("%d\t%d\n", fahr, celsius);
      fahr = fahr + step;
   }
}
```

O comando *for*

O Comando *For*

- Controle explícito das repetições de um laço

```
/* progPag024.c */
#include <stdio.h>
/* print Fahrenheit-Celsius table */
int main()
{
   int fahr;
   for (fahr = 0; fahr <= 300; fahr = fahr + 20)
      printf("%3d %6.1f\n", fahr, (5.0/9.0)*(fahr-32));
}
```

While x For

```c
#include <stdio.h>
int main()
{
   int fahr, celsius;
   int lower, upper, step;
   lower = 0;
   upper = 300;
   step = 20;
   fahr = lower;
   while (fahr <= upper)
   {
      celsius = 5 * (fahr-32) / 9;
      printf("%d\t%d\n", fahr, celsius);
      fahr = fahr + step;
   }
}
```

```c
#include <stdio.h>
/* print Fahrenheit-Celsius table */
int main()
{
   int fahr;
   for (fahr = 0; fahr <= 300; fahr = fahr + 20)
      printf("%3d %6.1f\n", fahr, (5.0/9.0)*(fahr-32));
}
```

Constantes simbólicas

Constantes Simbólicas

- Aumento da clareza do código

```
/* progPag026.c */

#include <stdio.h>
#define LOWER 0   /* lower limit of table */
#define UPPER 300 /* upper limit */
#define STEP 20   /* step size */

int main() /* print Fahrenheit-Celsius table */
{
   int fahr;
   for (fahr = LOWER; fahr <= UPPER; fahr = fahr + STEP)
      printf("%3d %6.1f\n", fahr, (5.0 / 9.0)*(fahr-32));
}
```

Resultado do Programa Fahrenheit Celsius

```c
#include <stdio.h>
#define LOWER 0 /* lower limit of table */
#define UPPER 300 /* upper limit */
#define STEP 20 /* step size */

main() /* print Fahrenheit-Celsius table */
{
   int fahr;
   for (fahr = LOWER; fahr <= UPPER; fahr = fahr + STEP)
      printf("%3d %6.1f\n", fahr, (5.0 / 9.0)*(fahr-32));
}
```

```
$ ./progPag026
  0  -17.8
 20   -6.7
 40    4.4
 60   15.6
 80   26.7
100   37.8
120   48.9
140   60.0
160   71.1
180   82.2
200   93.3
220  104.4
240  115.6
260  126.7
280  137.8
300  148.9
$
```

Exercício U1.1

- Escreva um programa para imprimir a tabela correspondente de Celsius para Fahrenheit
  - Valor inicial: -5 °C
  - Valor Final: 105 °C
  - Incremento: 10 °C

- Utilize o comando `while`

Exercício U1.2

- Escreva um programa para imprimir a tabela correspondente de Celsius para Fahrenheit
  - Valor inicial: -5 °C
  - Valor Final: 105 °C
  - Incremento: 10 °C

- Utilize o comando `for`

Uma coleção de programas úteis

Copia na Tela o Conteúdo de um Arquivo

```
Leia um caractere
Enquanto (caractere não é um EOF)
    Imprima caractere lido
    Leia novo caractere
```

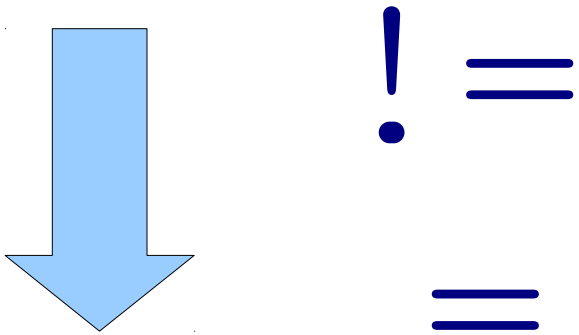
```
/* progPag027a.c */

#include <stdio.h>
/* copy input to output; 1st version */
int main()
{
    int c;
    c = getchar();
    while (c != EOF) /* Ctrl + D to leave the loop*/
    {
        putchar(c);
        c = getchar();
    }
}
```

Copia na Tela o Conteúdo de um Arquivo

```c
#include <stdio.h> /* progPag027a.c */
int main() /* copy input to output; 1st version */
{
   int c;
   c = getchar();
   while (c != EOF) /* Ctrl + D to leave the loop*/
   {
      putchar(c);
      c = getchar();
   }
}
```

```c
#include <stdio.h> /* progPag027b.c */
int main()   /* copy input to output; 2nd version */
{
   int c;
    while ((c = getchar())!= EOF) /* Ctrl + D to leave */
       putchar(c);
}
```

Um pouco sobre precedência de operadores




```
c = getchar()!= EOF

c = (getchar()!= EOF)

(c = getchar())!= EOF
```

Contagem de Caracteres

```
#include <stdio.h> /* progPag028.c */
int main()  /* count characters in input; 1st version */
{
   long nc;
   nc = 0;
   while (getchar() != EOF)
      ++nc;
   printf("\n%ld\n", nc);
}
```

long

++nc

%ld

Contagem de Caracteres

```c
#include <stdio.h> /* progPag028.c */
int main()  /* count characters in input; 1st version */
{
   long nc;
   nc = 0;
   while (getchar() != EOF)
      ++nc;
   printf("\n%ld\n", nc);
}
```

```c
#include <stdio.h> /* progPag029a.c */
int main() /* count characters in input; 2nd version */
{
   double nc;
   for (nc = 0; getchar() != EOF; ++nc)
      ;
   printf("\n%.0f\n", nc);
}
```

Contagem de Linhas

```c
#include <stdio.h> /* progPag029b.c */
int main() /* count lines in input */
{
   int c, nl;

   nl = 0;
   while ((c = getchar()) != EOF)
      if (c == '\n')
         ++nl;
   printf("\n%d\n", nl);
}
```

==

'c'

"c"

'\n'

Regras de Tratamento do Tipo `char`

`'A'`	Constante char
`'\n' '\t' '\b'`	Caracteres especiais
`'A' ≡ 65`	int e char

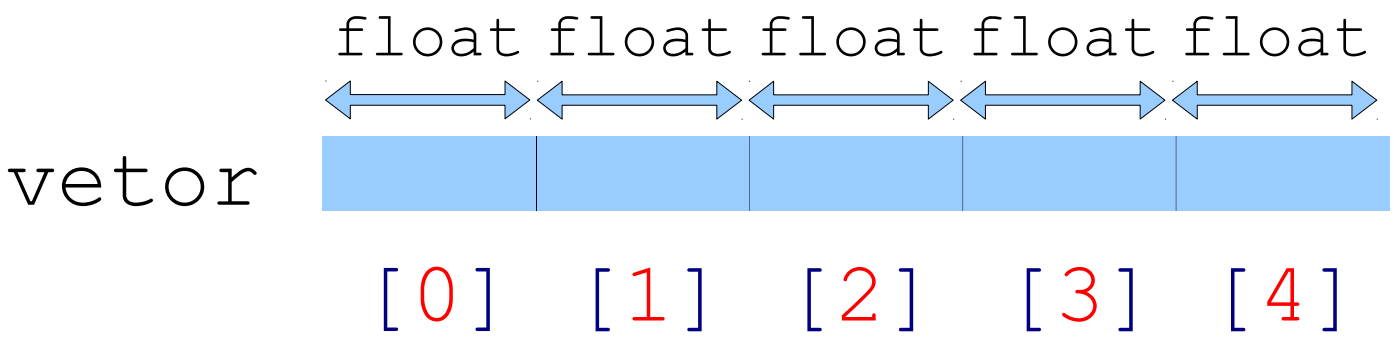
Contagem de Palavras

```c
#include <stdio.h> /* progPag030.c */
#define IN  1 /* inside a word */
#define OUT 0 /* outside a word */

main() /* count lines, words, and characters in input */
{
   int c, nl, nw, nc, state;
   state = OUT;
   nl = nw = nc = 0;

   while ((c = getchar()) != EOF)
   {
      ++nc;
      if (c == '\n')
         ++nl;
      if (c == ' ' || c == '\n' || c == '\t')
         state = OUT;
      else if (state == OUT)
      {
         state = IN;
         ++nw;
      }
   }
   printf("\nLFs %d Words %d Characters %d\n", nl, nw, nc);
}
```

Arrays

Arrays

```
float vetor[5];
```

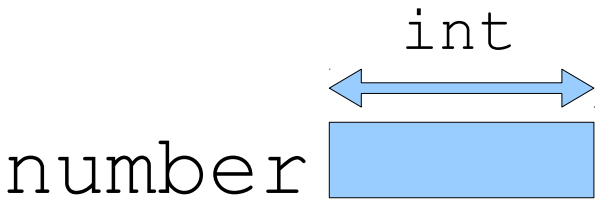


Arrays

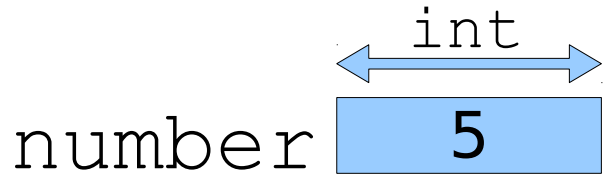
```
#include <stdio.h> /* progPag032.c */
int main()/* count digits, white space, others */
{
   int c, i, nwhite, nother;
   int ndigit[10];
   nwhite = nother = 0;

   for (i = 0; i < 10; ++i)
      ndigit[i] = 0;

   while ((c = getchar()) != EOF)
      if (c >= '0' && c <= '9')
         ++ndigit[c-'0'];
      else if (c == ' ' || c == '\n' || c == '\t')
         ++nwhite;
      else
         ++nother;

   printf("\ndigits =");

   for (i = 0; i < 10; ++i)
      printf(" %d", ndigit[i]);

   printf(", white space = %d, other = %d\n", nwhite, nother);
}
```

Alocação automática de memória

```
int number;
```

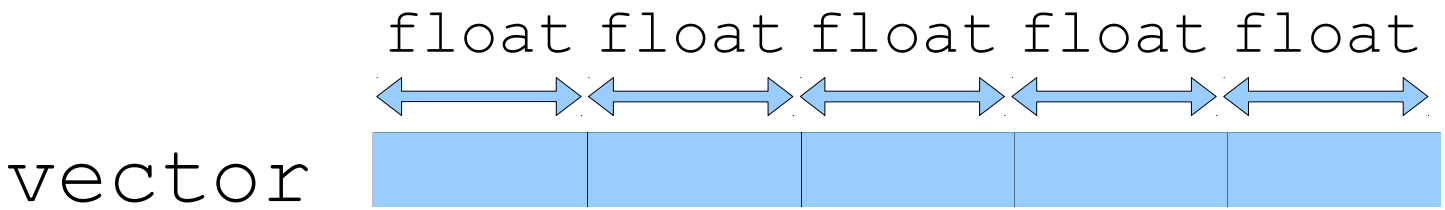



```
int number = 5;
```

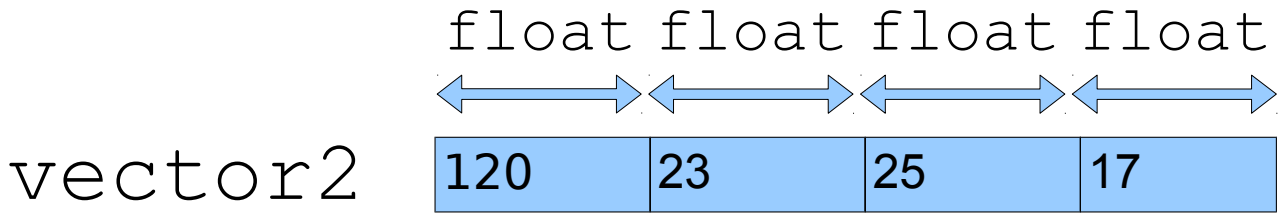


Alocação automática de memória

```
float vector[5];
```




```
float vector2[]={120, 23, 25, 17};
```



Exercício U1.3

- Escreva um programa que conte o número de ocorrências de cada dígito numérico. Seu programa deve contar também o números total de dígitos não numéricos. Ao final o programa deve calcular o histograma de probabilidades de cada dígito e do grupo de outros caracteres

Funções

Funções

```
#include <stdio.h> /* progPag034.c */
int power(int m, int n);

int main()
/* test power function */
{
   int i;

   for (i = 0; i < 10; ++i)
      printf("%d %d %d\n", i, power(2,i), power(-3,i));

   return 0;
}

/* power: raise base to n-th power; n >= 0 */
int power(int base, int n)
{
   int i, p;
   p = 1;

   for (i = 1; i <= n; ++i)
      p = p * base;
   return p;
}
```

Argumentos: Chamada por Valor

Argumentos: Chamada por Valor

```
/* progPag035.c */
/* power: raise base to n-th power; n >= 0 */
int power(int base, int n)
{
   int i, p;
   p = 1;

   for (i = 1; i <= n; ++i)
      p = p * base;
   return p;
}
```

REGRAS:
Na linguagem C somente existe passagem de parâmetros por valor.
Os parâmetros de uma função são alocados de forma automática.

Array de char
=
String

Array de char = String

```
#include <stdio.h> /* progPag037.c */
#define MAXLINE 1000 /* maximum input line length */
int mygetline(char line[], int maxline);
void copy(char to[], char from[]);

int main()/* print the longest input line */
{
   int len; /*current line length */
   int max; /*maximum length seen so far */
   char line[MAXLINE]; /* current input line */
   char longest[MAXLINE]; /* longest line saved here */

   max = 0;
   while ((len = mygetline(line, MAXLINE)) > 0)
      if (len > max)
      {
         max = len;
         copy(longest, line);
      }

   if (max > 0) /* there was a line */
      printf("%s", longest);
   return 0;
}
```

Array de char = String

```
/* progPag037.c getline: read a line into s, return length */
int mygetline(char s[],int lim)
{
   int c, i;
   for (i=0; i < lim-1 && (c=getchar())!=EOF && c!='\n'; ++i)
      s[i] = c;
   if (c == '\n')
   {
      s[i] = c;
      ++i;
   }
   s[i] = '\0';
   return i;
}

/* copy: copy 'from' into 'to'; assume to is big enough */
void copy(char to[], char from[])
{
   int i;
   i = 0;

   while ((to[i] = from[i]) != '\0')
      ++i;
}
```

Variáveis Externas

Variáveis Externas

progPag039.h

```
#define MAXLINE 1000 /* maximum input line size */

int max; /* maximum length seen so far */
char line[MAXLINE]; /* current input line */
char longest[MAXLINE]; /* longest line saved here */
int mygetline(void);
void copy(void);
```

Variáveis Externas

progPag039.c

```c
#include <stdio.h>
int main() /* print longest input line; specialized version */
{
   int len;
   extern int max;
   extern char longest[];
   max = 0;
   while ((len = mygetline()) > 0)
   if (len > max)
   {
      max = len;
      copy();
   }
   if (max > 0) /* there was a line */
      printf("%s", longest);
   return 0;
}
```

Variáveis Externas

progPag039.c

```
/* progPag039.c */
/* getline: specialized version */
int mygetline(void)
{
   int c, i;
   extern char line[];

   for (i=0; i < MAXLINE-1 && (c=getchar())!=EOF && c != '\n'; ++i)
      line[i] = c;
   if (c == '\n')
   {
      line[i] = c;
      ++i;
   }
   line[i] = '\0';
   return i;
}
```

Variáveis Externas

progPag039.c

```
/* progPag039.c */
/* copy: specialized version */
void copy(void)
{
   int i;
   extern char line[], longest[];
   i = 0;

   while ((longest[i] = line[i]) != '\0')
      ++i;
}
```

Trabalho Final do Capítulo 1

- Escreva um programa que leia um texto com quantidade de colunas ilimitado e produza na tela uma versão deste mesmo texto com largura máxima de 80 colunas.

  Dica: a cada nova palavra lida deve ser avaliado se esta fará com que a linha atual exceda 80 colunas.

FIM