DEI
DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
TÉCNICO LISBOA

Introdução à Programação em C

Input/Output

IAED

Ponto da situação

- O programa começa sempre por executar a função **`main()`**
- Inclusão de funções adicionais ao C através da directiva `#include`
- Definição de constantes usando directivas de pré-compilador `#define`
- Tipos de dados básicos: **`char`**, **`int`**, **`float`** e **`double`**
- Ciclos utilizando a estrutura de controlo **`while`**
- Ciclos utilizando a estrutura de controlo **`for`**

Input & Output

- Já sabemos escrever para o terminal através do comando **`printf`** e ler do terminal através do comando **`scanf`**.

Outro desafio: Contagem de valores

- **Problema**
  - Contar uma lista de inteiros positivos introduzidos pelo utilizador (por exemplo, lista de notas de uma disciplina)
  - Qualquer valor negativo determina o fim da lista de inteiros a ler

Input

```
3
18
15
10
12
17
-1
```

Output

```
6
```

Outro desafio: Contagem de valores

- **Mas antes disso... Para ver se tudo bate certo:**
  - Vamos ler a lista de inteiros positivos introduzidos pelo utilizador (por exemplo, lista de notas de uma disciplina) e mostramos os valores lidos no ecrã
  - Qualquer valor negativo determina o fim da lista de inteiros a ler

Input

```
3
18
15
10
12
17
-1
```

Output

```
3
18
15
10
12
17
```

Cópia de Inteiros Positivos

- **Mas antes disso... Para ver se tudo bate certo:**
  - Vamos ler a lista de inteiros positivos introduzidos pelo utilizador (por exemplo, lista de notas de uma disciplina) e mostramos os valores lidos no ecrã
  - Qualquer valor negativo determina o fim da lista de inteiros a ler

- Funções de manipulação do standard input e output
  - `scanf()` leitura formatada do input
  - `printf()` escreve texto formatado no output

Cópia de Inteiros Positivos

Algoritmo

Lê um inteiro e guarda-o numa variável "v"

Enquanto v for válido

- Escreve o inteiro lido
- Lê próximo inteiro e guarda-o novamente na variável "v"

Cópia de Inteiros Positivos

Algoritmo

```
Lê um inteiro e guarda-o numa variável "v"
scanf("%d", &v);
Se v for válido
while (v >= 0) {
    Escreve o inteiro lido
    printf("%d\n", v);
    Lê próximo inteiro e guarda-o novamente na variável "v"
    scanf("%d", &v);
}
```

Cópia de Inteiros Positivos

```
#include <stdio.h>

/* Copia inteiros do input para output */

int main ()
{
  int v;

  scanf("%d", &v);
  while (v >= 0){
    printf("%d\n", v);
    scanf("%d", &v);
  }
  return 0;
}
```

- Operadores de comparação
  - Igualdade: ==
  - Desigualdade: > < >= <= !=

Cópia de Inteiros Positivos

```
scanf("%d", &v);
while (v >= 0){
  printf("%d\n", v);
  scanf("%d", &v);
}
```

- %d para ler um inteiro, %f para um float, etc.

```
scanf("%d", &v);
```

- Necessário colocar **&** antes do nome da variável

Contagem do Tamanho de Lista

- Outro problema:
  - Tal como no exemplo anterior, vamos ler uma lista de inteiros positivos introduzidos pelo utilizador (por exemplo, lista de notas de uma disciplina)
  - Qualquer valor negativo determina o fim da lista de inteiros a ler
  - **No fim mostrar o número de elementos que compõem a lista**

Contagem do Tamanho de Lista

Algoritmo

Inicializa **contador** a 0

Lê um inteiro

Enquanto conseguir ler um valor válido

- Incrementa **contador**
- Lê próximo inteiro

Escreve valor do contador

Contagem do Tamanho de Lista

Algoritmo

```
Inicializa contador a 0
contador = 0;
Lê um inteiro
scanf("%d", &v);
Enquanto conseguir ler um valor válido
while (v >= 0){
    Incrementa contador
    ++contador;
    Lê próximo inteiro
    scanf("%d", &v);
}
Escreve valor do contador
printf("%ld\n", contador);
```

Contagem do Tamanho de Lista

```
#include <stdio.h>

/* Conta número de elementos de lista de números positivos */

int main ()
{
  int v;
  long contador;

  contador = 0;
  scanf("%d", &v);
  while (v >= 0) {
     ++contador;
     scanf("%d", &v);
  }
  printf("%ld\n", contador);
  return 0;
}
```

long?

++ ?

Contagem do Tamanho de Lista

```
#include <stdio.h>

int main ()
{
  int v;
  long contador;

  contador = 0;
  scanf("%d", &v);
  while (v >= 0) {
     ++contador;
     scanf("%d", &v);
  }
  printf("%ld\n", contador);
  return 0;
}
```

- Variável do tipo **`long int`**
- Pelo menos 32 *bits* (4 *bytes*) para guardar o inteiro
- `%ld` para escrever com `printf`

Valores de $-(2^{31}-1)$ a $(2^{31}-1)$

Contagem do Tamanho de Lista

```
#include <stdio.h>

int main ()
{
  int v;
  long contador;

  contador = 0;
  scanf("%d", &v);
  while (v >= 0) {
     ++contador;
     scanf("%d", &v);
  }
  printf("%ld\n", contador);
  return 0;
}
```

- Incremento da variável `contador`
- Neste caso é igual a **`contador = contador+1`** `ou` **`contador+=1`**
- `contador++ , --contador , contador--`

++ e -- : forma prefixa / sufixa

```
int a = 0, b = 0, c = 0;

a++;
++a;
```

O Resultado é o mesmo!

```
int a = 0;

while (a++ <= 3) {
  ...                          /* O ciclo é executado 4 vezes */
}
```

Soma 1 a "a" depois de testar a<=3

```
int a = 0;

while (++a <= 3) {
  ...                          /* O ciclo é executado 3 vezes */
}
```

Soma 1 a "a" antes de testar a<=3

- A utilização dos operadores ++ e -- de forma prefixa ou sufixa pode influenciar a execução do programa.

++ e -- : forma prefixa / sufixa (exercício)

```
i = 1;
j = ++i;

printf("%d %d\n", i, j);
```

Qual o output?

```
i = 1;
j = i++;

printf("%d %d\n", i, j);
```

Qual o output?

Contagem do Tamanho de Lista

`for` *vs* `while`

```
contador = 0;
while (v >= 0) {
   scanf("%d", &v);
   ++contador;
}
```

```
for (contador = 0; v >= 0; ++contador)
    scanf("%d", &v);
```

- Estrutura de controlo `for` para ciclos contados
- Inicialização de variáveis: `contador = 0`
- Teste: `v >= 0`
- Incremento: `++contador`

Contagem do Tamanho de Lista

```
#include <stdio.h>

/* Conta número de elementos de lista de números positivos */

int main ()
{
  int v;
  long contador;

  scanf("%d", &v);
  for (contador = 0; v >= 0; ++contador)
    scanf("%d", &v);
  printf("%ld\n", contador);
  return 0;
}
```

Contagem de Aprovações e Reprovações

- **2º problema de hoje**:
  - Considere-se uma lista de inteiros que denota as notas dos alunos numa disciplina
  - Ler uma lista de inteiros positivos introduzidos pelo utilizador
  - Qualquer valor negativo determina o fim do conjunto de inteiros a ler
  - **No fim mostrar a seguinte informação:**
    - **número de notas à disciplina**
    - **número de aprovações**
    - **número de reprovações**

Contagem de Aprovações e Reprovações

Algoritmo

Inicializa contador de notas e aprovações a 0

Lê nota

Enquanto conseguir ler uma nota válida

    Incrementa o contador de notas

    Se a nota lida é positiva

        Incrementa contador de aprovações

    Lê nota seguinte

Escreve valor dos contadores

Contagem de Aprovações e Reprovações

Algoritmo

```
Inicializa contador de notas e aprovações a 0
notas = aprovacoes = 0;
Lê nota
scanf("%d", &v);
Enquanto conseguir ler uma nota válida
while (v >= 0) {
        Incrementa o contador de notas
        notas++;
        Se a nota lida é positiva
        if (v >= 10)
                Incrementa contador de aprovações
                aprovacoes++;
        Lê nota seguinte
        scanf("%d", &v);
}
Escreve valor dos contadores
printf(…);
```

Contagem de Aprovações e Reprovações

```
#include <stdio.h>

/* Contagem de Aprovações e Reprovações de lista de notas */

#define NOTA_MIN_APROVACAO 10

int main () {
  int v, notas, aprovacoes;

  notas = aprovacoes = 0;
  scanf("%d", &v);
  while (v >= 0) {
    notas++;
    if (v >= NOTA_MIN_APROVACAO)
      aprovacoes++;
    scanf("%d", &v);
  }
  printf("Total: %d, Aprovacoes: %d, Reprovacoes: %d\n",
         notas, aprovacoes, notas-aprovacoes);
  return 0;
}
```

Contagem de Aprovações e Reprovações

```
int main ()
{
  int v, notas, aprovacoes;

  notas = aprovacoes = 0;
  scanf("%d", &v);
  while (v >= 0)
  {
    ...
  }
  printf("Total: %d, Aprovacoes: %d, Reprovacoes: %d\n",
         notas, aprovacoes, notas-aprovacoes);
  return 0;
}
```

- Atribuição múltipla
- Equivale a `(notas = (aprovacoes = 0));`

Exercício 1: Maior Nota na Lista

- Modifique o programa anterior por forma a calcular também qual a maior nota presente na lista

Exercício 2: Média das Positivas

- Modifique o programa anterior por forma a calcular a média das notas positivas

Linha de comandos

- Como fornecer um input guardado num ficheiro a um programa através do stdin?

```
$ ./myprogram < input.txt
```

- O output também pode ser guardado num ficheiro através do operador ">"

```
$ ./myprogram > output.txt
```

- Os dois operadores podem ser combinados

```
$ ./myprogram < input.txt > output.txt
```

Input/Output de Texto

- Streams de Texto
  - Leitura e escrita de caracteres
  - Caracteres como números inteiros
- Exemplos
  - Leitura e Escrita de Caracteres
  - Contagem de Caracteres
  - Contagem de Linhas
  - Contagem de Palavras

Leitura e Escrita de Caracteres

- *Text stream:* sequência de caracteres
  - entrada
  - saída
- Podem corresponder a leitura/escrita de ficheiros ou leitura/escrita do terminal (*standard input/output*)
- Cada linha contém 0 ou mais caracteres e acaba com o caracter `\n`
- Funções de manipulação de *text streams (stdout/stdin)*
  - ***`getchar()`*** lê o proximo caracter da *text stream*
  - ***`putchar(n)`*** escreve o caracter *c* cujo código ASCII é o número inteiro (?) passado como argumento

Caracteres e inteiros em C: Qual a diferença?

```
#include <stdio.h>

int main()
{
    char a ='T';
    printf("Ao caracter %C corresponde o ASCII %d\n", a , a );
    return 0;
}
```

- Resultado:

```
Ao caracter T corresponde o ASCII 84
```

- As variáveis do tipo char são (pequenos) inteiros (1 Byte).
- Podemos realizar operações numéricas tal como fazemos com os `int`'s.

Tabela ASCII

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(	72	48	H	104	68	h
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	□

Imagem retirada de:

http://www.cdrummond.qc.ca/cegep/informat/Professeurs/Alain/files/ascii.htm

Algarismos

letras

LETRAS

Leitura e Escrita de Caracteres

– Ler um texto, carácter a carácter, e imprimir no ecran o que se vai lendo.

Funções de manipulação do standard input e output

- `getchar` leitura de 1 char
- `putchar` escrita de 1 char

Leitura e Escrita de Caracteres

Algoritmo

Lê um caracter

Enquanto o carácter não for o de fim de ficheiro

Escreve caracter

Lê próximo caracter

Leitura e Escrita de Caracteres

Algoritmo

Lê um caracter

```
c = getchar();
```

Enquanto o caracter não for o de fim de ficheiro

```
while (c != EOF) {
```

Escreve caracter

```
    putchar(c);
```

Lê próximo caracter

```
    c = getchar();
}
```

Leitura e Escrita de Caracteres

```
#include <stdio.h>

/* Copia input para output */

int main ()
{
  int c;

  c = getchar();
  while (c != EOF) {
    putchar(c);
    c = getchar();
  }
  return 0;
}
```

- != significa diferente
- Constante `EOF = End Of File` (no terminal unix = Crtl-D)

Leitura e Escrita de Caracteres

```
c = getchar();
while (c != EOF){
  putchar(c);
  c = getchar();
}
```

```
while ((c = getchar()) != EOF)
  putchar(c);
```

- Precedência de `!=` maior do que `=`

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

- Necessários parênteses em:

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

Contagem de Caracteres

Algoritmo

Inicializa contador a 0

Enquanto o caracter lido não for o de fim de ficheiro

Incrementa contador

Escreve valor do contador

Contagem de Caracteres

Algoritmo

```
Inicializa contador a 0
contador = 0;
Enquanto o caracter lido não fôr o de fim de ficheiro
while (getchar() != EOF)
    Incrementa contador
    ++contador;
Escreve valor do contador
printf("%ld\n", contador);
```

Contagem de Caracteres

```
#include <stdio.h>

/* Conta caracteres do input */

int main ()
{
  long contador;

  contador = 0;
  while (getchar() != EOF)
     ++contador;
  printf("%ld\n", contador);
  return 0;
}
```

Contagem de Linhas

Algoritmo

Inicializa contador a 0

Enquanto o carácter lido não for o de fim de ficheiro

Se o carácter lido for o de fim de linha então

Incrementa contador

Escreve valor do contador

Contagem de Linhas

Algoritmo

Inicializa contador a 0

```
contador = 0;
```

Enquanto o carácter lido não for o de fim de ficheiro

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

Se o carácter lido for o de fim de linha então

```
    if (c == '\n')
```

Incrementa contador

```
        ++contador;
```

Escreve valor do contador

```
printf("%d\n",contador);
```

Contagem de Linhas

```
#include <stdio.h>

/* Conta linhas do input */

int main ()
{
  int c, contador;

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

Contagem de Linhas

```
#include <stdio.h>

/* Conta linhas do input */

int main ()
{
  int c, contador;

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

- Teste de igualdade `==`
- Código ASCII do caracter de mudança de linha `'\n'`

Exercício 1: Contagem de Algarismos

- Construa um programa que lê caracteres do input e conta o número de algarismos.

Tabela ASCII

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(	72	48	H	104	68	h
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	□

Imagem retirada de:

http://www.cdrummond.qc.ca/cegep/informat/Professeurs/Alain/files/ascii.htm

Algarismos

letras

LETRAS

Exercício 1: Contagem de Algarismos

```
#include <stdio.h>

/* Conta algarismos do input */

int main ()
{
    int c;
    long contador;

    contador = 0;

    while ((c = getchar()) != EOF)
        if (c >= '0' && c<='9')
            contador++;

    printf("%ld algarismos\n",contador);

    return 0;
}
```

Exercício 2: Contagem de Letras

- Construa um programa que lê caracteres do input e conta o número de letras minúsculas e maiúsculas no texto.

Exercício 3: Maiúsculas para Minúsculas

- Construa um programa que lê caracteres do input e mostra o mesmo texto no output, mas alterando as letras maiúsculas para letras minúsculas.

Ex. 4 (mais complicado): Contagem de Palavras

- Construa um programa que lê caracteres do input e conta o número de palavras no texto.

Contagem de Palavras

Algoritmo

Inicializa estado a FORA

Inicializa contador de caracteres e palavras a 0

Enquanto o carácter lido não for o de fim de ficheiro

    Incrementa contador de caracteres

    Se o carácter lido for o de fim de linha, espaço ou tabulação então

        Estado toma valor FORA

    Caso contrário e se o estado for FORA então

        Estado toma valor DENTRO

        Incrementa contador de palavras

Escreve valor dos contadores

Contagem de Palavras

Algoritmo

```
Inicializa estado a FORA
estado = FORA;
Inicializa contador de letras e palavras a 0
np = nc = 0;
Enquanto o carácter lido não for o de fim de ficheiro
while ((c = getchar()) != EOF) {
    Incrementa contador de letras
    ++nc;
    Se o carácter lido for o de fim de linha, espaço ou tabulação então
    if (c == ' ' || c == '\n' || c == '\t')
        Estado toma valor FORA
        estado = FORA;
    Caso contrário e se o estado for FORA então
    else if (estado == FORA) {
        Estado toma valor DENTRO
        estado = DENTRO;
        Incrementa contador de palavras
        ++np; } }
Escreve valor dos contadores
printf("%d %d\n", np, nc);
```

Contagem de Palavras

```
#include <stdio.h>
#define FORA 0
#define DENTRO 1

int main ()
{
  int c, np = 0, nc = 0, estado = FORA;

  while ((c = getchar()) != EOF) {
    ++nc;
    if (c == ' ' || c == '\n' || c == '\t')
      estado = FORA;
    else if (estado == FORA) {
      estado = DENTRO;
      ++np;
    }
  }
  printf("%d %d\n", np, nc);
  return 0;
}
```

Contagem de Palavras

```
int main ()
{
  int c, np = 0, nc = 0, estado = FORA;

  while ((c = getchar()) != EOF) {
    ++nc;
    ...
  }
```

- Inicialização de variáveis na declaração
- Variáveis não inicializadas podem ter um valor qualquer

Contagem de Palavras

```
    if (c == ' ' || c == '\n' || c == '\t')
      estado = FORA;
    else if (estado == FORA) {
      estado = DENTRO;
      ++np;
    }
  }
```

- Operador lógico disjunção ||
- Operador lógico conjunção && (maior precedência que ||)
- Argumentos avaliados da esquerda para direita
- Interrompe avaliação quando valor do argumento for suficiente para definir valor da expressão.

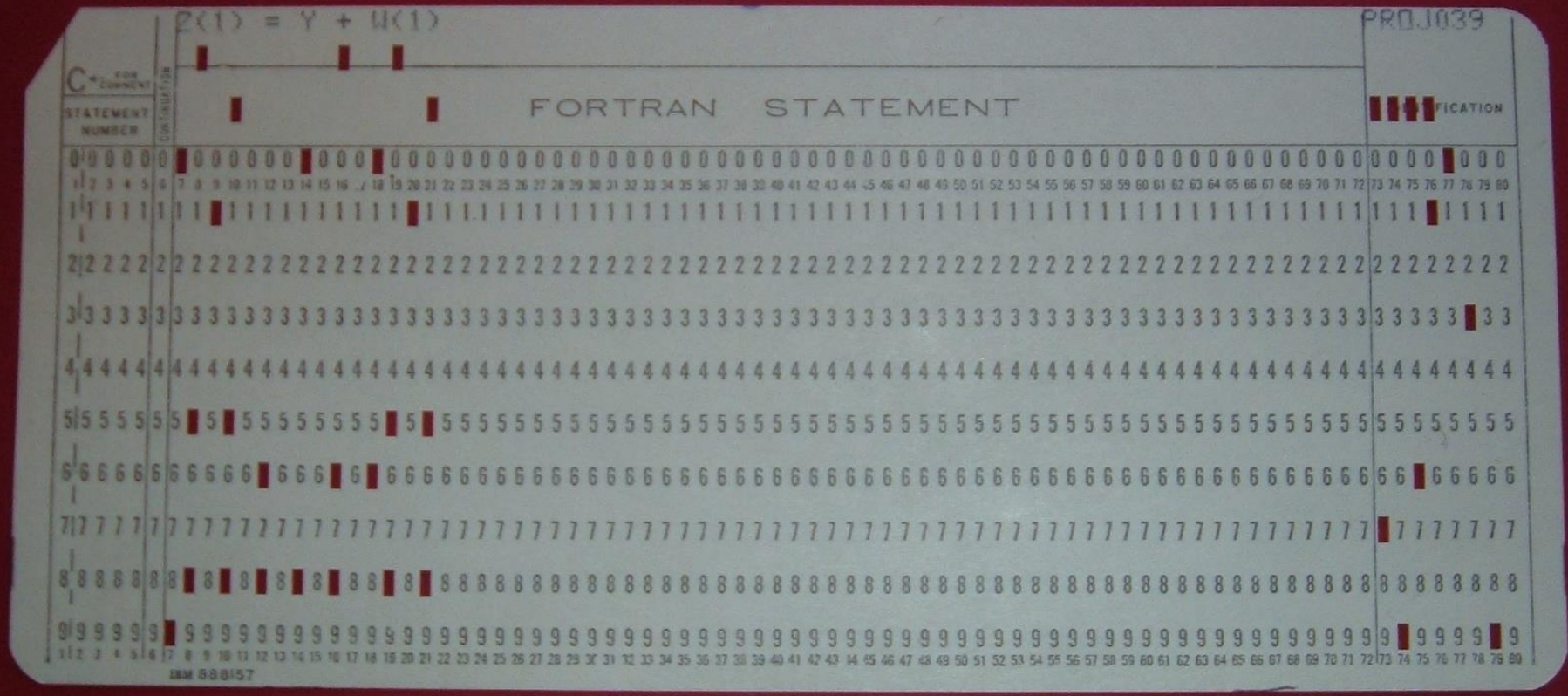
Contagem de Palavras

```
    if (c == ' ' || c == '\n' || c == '\t')
      estado = FORA;
    else if (estado == FORA) {
      estado = DENTRO;
      ++np;
    }
  }
```

- Instrução if-then-else
- Síntaxe:

```
if (<expressão>)
  <instrução>
else
  <instrução>
```

Cartão perfurado com uma linha de programa



Alguma Dúvida ?