DEI
DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
TÉCNICO LISBOA

Introdução à Programação em C

Tabelas

IAED

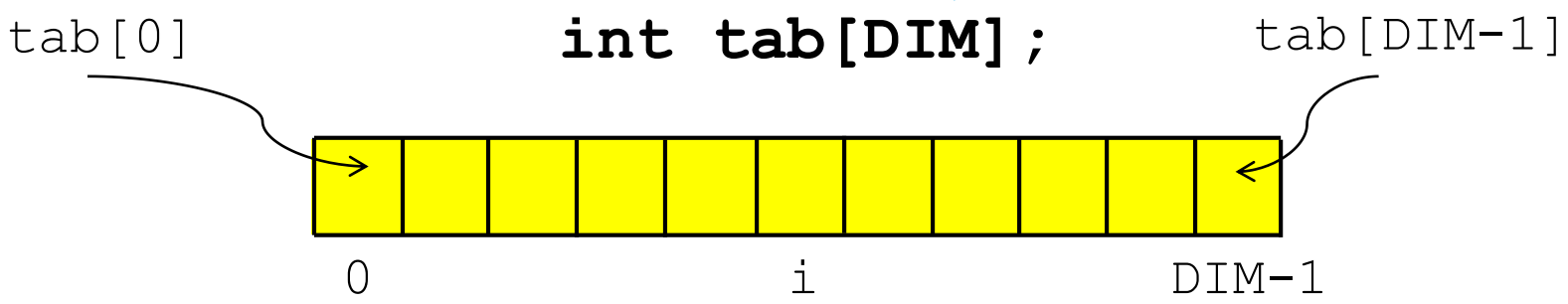
Resumo

- Parte I: Tabelas para guardar conjuntos de elementos
  - Tabelas Unidimensionais (vectores)
  - Tabelas Bidimensionais (matrizes)
  - Exemplos: tabelas de inteiros
- Parte II: Strings (cadeias de caracteres)
  - Exemplos: ler strings, copiar strings

Tabelas Unidimensionais (Vectores)

- Colecção de elementos
  - Inteiros, reais, caracteres

"ANSI C": Tem de ser uma constante !!



- `tab[i]`
  - Valor do inteiro na posição de índice `i`
- `tab[i]=10;`
  - Atribui o valor `10` ao inteiro na posição de índice `i`
- primeiro índice: `0` ; último índice: `DIM-1`
  - Programador é responsável por respeitar os limites!

Tabelas Unidimensionais (Vectores)

exemplo:

```
#include <stdio.h>

int main ()
{
  int i;
  int tab[10];

  for (i = 0; i < 10; i++)
    tab[i] = 2*i;

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

  return 0;
}
```

declaração

atribuição

Escrita formatada

Tabelas Unidimensionais (Vectores)

- E se pretendermos fazer uma leitura formatada de **`DIM`** valores, guardando informação directamente num vector?

```
#include <stdio.h>

#define DIM 10

int main ()
{
  int i, tab[DIM];

  for (i = 0; i < DIM ; i++)
    scanf("%d", &tab[i]);

  (…)
}
```

Podemos declarar tudo na mesma linha

O `scanf` é utilizado tal como anteriormente...

Exemplo 1: Contagem de Aprovações e ocorrências de Notas

- Objectivo:
  - 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
    - Número de ocorrências de cada uma das notas entre 0 e 20

A única coisa nova...

...por outras palavras: temos de fazer um histograma das notas

Contagem de Aprovações e Ocorrências de Notas

Algoritmo

... i.e, o nosso "vector" histograma

Inicializa ***contador de notas*** e de aprovações a 0

Lê nota

Enquanto conseguir ler uma nota válida

- Incrementa o contador de notas e da nota correspondente
- Se a nota lida é positiva
  - Incrementa contador de aprovações
- Lê nota seguinte

Escreve valor dos contadores

Contagem de Aprovações e Ocorrências de Notas

Algoritmo

```
Inicializa o histograma de notas e contador de aprovações e reprovações a 0
aprovacoes = total = 0;
for (i=0; i<21; i++)
   histograma[i]=0;
Lê nota
scanf("%d", &v);
Enquanto conseguir ler uma nota válida
while (v >= 0) {
        Incrementa o contador da nota correspondente
        total++;
        Incrementa o contador da nota correspondente
        histograma[v]++;
        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 ("Total: %d, Aprovacoes: %d, Reprovacoes: %d\n",
   total, aprovacoes, total - aprovacoes);

for (i=0; i<21; i++)
   printf ("%d %d\n", i, histograma[i]);
```

Contagem de Aprovações e Ocorrências de Notas

```
#include <stdio.h>
#define DIM 21
#define NOTA_MINIMA 10
int main () {
  int i, v, total = 0, aprovacoes = 0, histograma[DIM];
  for (i = 0; i < DIM; i++)
      histograma[i] = 0 ;

  scanf("%d", &v);
  while (v >= 0) {
    total++;
    histograma[v]++;
    if (v >= NOTA_MINIMA)
      aprovacoes++;
    scanf("%d", &v);
  }

   printf ("Total: %d, Aprovacoes: %d, Reprovacoes: %d\n", total,
            aprovacoes, total-aprovacoes);
   for (i = 0; i < DIM; i++)
        printf ("%d %d\n", i, histograma[i]);
   return 0;
}
```

Contagem de Aprovações e Ocorrências de Notas

```
#define DIM 21

int main () {
  int i, v, total = 0, aprovacoes = 0, histograma[DIM];

  for (i = 0; i < DIM; i++)
      histograma[i] = 0;
```

- Declaração de tabelas unidimensionais
- Variável `histograma` é tabela de 21 inteiros (`DIM` é a dimensão da tabela)
- `histograma[0]`, ..., `histograma[DIM-1]`
- `histograma[i]` é o inteiro no índice `i` da tabela `histograma`
- `histograma[i]=0` atribui o valor `0` ao inteiro na posição `i`

Exemplo 2: Número de Aprovações e Nota Mais Alta de um Turno

- Objectivo:
  - Considere-se uma lista de inteiros que denota as notas dos alunos inscritos em **1 turno** prático com **25 alunos** (assume-se que o tamanho do turno é fixo e o turno está completo)
  - Ler uma lista de 25 inteiros positivos introduzidos pelo utilizador e **guardar esta informação num vector de dimensão 25.**
  - Usando a informação guardada no vector, calcular:
    - Número de aprovações
    - Nota mais alta
  - Mostrar o resultado.

Número de Aprovações e Nota Mais Alta de um Turno

Algoritmo

Inicializa contador de notas, aprovações e nota mais alta a 0
Enquanto contador de notas inferior a 25
    Lê nota e guarda na tabela
    Incrementa contador de notas

Inicializa contador de notas a 0
Enquanto contador de notas inferior a 25
    Se a nota é positiva
        Incrementa contador de aprovações
    Se a nota é maior que a nota mais alta guardada
        Atribui o valor da nota lida à nota mais alta
    Incrementa contador de notas
Escreve número de aprovações e nota mais alta

Número de Aprovações e Nota Mais Alta de um Turno

Número de alunos por turno

```
#include <stdio.h>
#define DIM 25
#define NOTA_MINIMA 10

int main () {
  int i, aprovacoes = 0, alta = 0, notas[DIM];

  for (i = 0; i < DIM; i++)
    scanf("%d", &notas[i]);

  for (i = 0; i < DIM; i++) {
    if (notas[i] >= NOTA_MINIMA)
      aprovacoes++;
    if (notas[i] > alta)
      alta = notas[i];
  }

  printf("Aprovacoes: %d, Mais alta: %d\n", aprovacoes, alta);

  return 0;
}
```

Leitura das 25 notas do turno

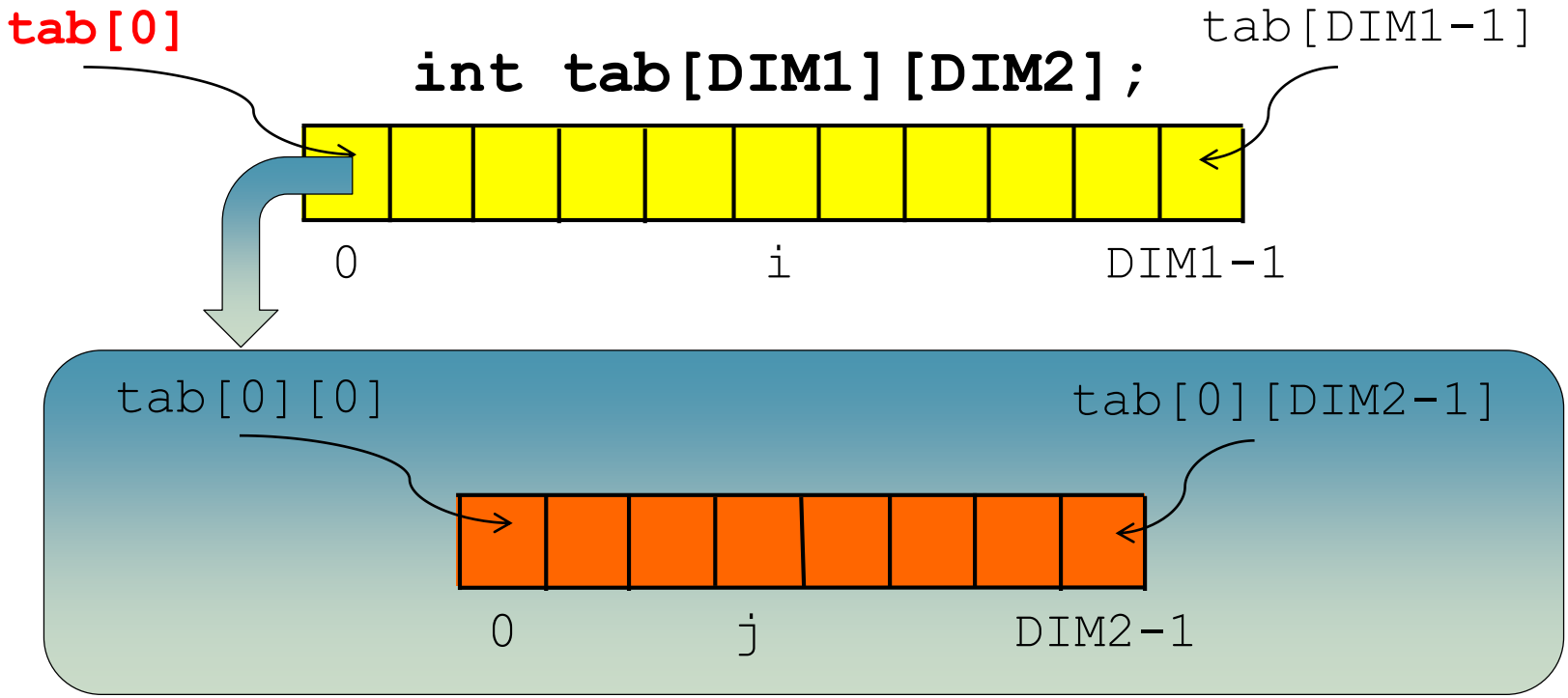
Cálculo do número de aprovações e nota mais alta

Exemplo 3: Número de Aprovações e Nota Mais Alta num Conjunto de Turnos

Ou seja, temos um vector de turnos

- Objectivo:
  - Considere-se os **4 turnos práticos** de uma disciplina e 4 listas de inteiros que denotam as notas dos alunos inscritos **em cada um** dos turnos práticos (assume-se que o tamanho do turno é fixo e igual a **25 alunos** e todos os turnos estão completos)
  - **Para cada turno** ler uma lista de 25 inteiros positivos introduzidos pelo utilizador
  - No fim mostrar a seguinte informação **para cada turno**:
    - Número de aprovações
    - Nota mais alta

Tabelas Bidimensionais (Matrizes)



- `tab[i]` é o vector de inteiros no índice `i` da tabela
- `tab[i][j]` é o inteiro no índice `j` da posição `i` da tabela (linha `i`, coluna `j`)
- `tab[i][j]=10` atribui o valor `10` ao inteiro no índice `j` da posição `i` da tabela

Exemplo 3: Número de Aprovações e Nota Mais Alta num Conjunto de Turnos

Algoritmo

Inicializa contador de turnos a 0
Enquanto contador de turnos inferior a 4
    Lê e guarda na tabela as notas de um turno
    Incrementa contador de turnos

Inicializa contador de turnos a 0
Enquanto contador de turnos inferior a 4
    Inicializa contador de aprovações do turno a 0
    Inicializa nota mais alta do turno a 0
    Calcula e guarda número de aprovações do turno
    Calcula e guarda a nota mais alta do turno
    Incrementa o contador de turnos

Escreve número de aprovações e nota mais alta de cada turno

Exemplo 3: Número de Aprovações e Nota Mais Alta num Conjunto de Turnos

```
#define TURNOS 4
#define ALUNOS 25

int main () {
  int i, j, aprovacoes[TURNOS], alta[TURNOS], notas[TURNOS][ALUNOS];









  return 0;
}
```

#aprovacões em cada turno

Nota mais alta para cada turno

Todas as notas de todos os turnos

`notas[i][j]` = nota do aluno `j` do turno `i`

Exemplo 3: Número de Aprovações e Nota Mais Alta num Conjunto de Turnos

```
#include <stdio.h>
#define TURNOS 4
#define ALUNOS 25
#define NOTA_MINIMA 10

int main () {
  int i, j, aprovacoes[TURNOS], alta[TURNOS], notas[TURNOS][ALUNOS];

  for (i = 0; i < TURNOS; i++)
    for (j = 0; j < ALUNOS; j++)
      scanf("%d", &notas[i][j]);

  for (i = 0; i < TURNOS; i++) {
    aprovacoes[i] = alta[i] = 0;
    for (j = 0; j < ALUNOS; j++) {
      if (notas[i][j] >= NOTA_MINIMA)
        aprovacoes[i]++;
      if (notas[i][j] > alta[i])
        alta[i] = notas[i][j];
    }
  }
  for (i = 0; i < TURNOS; i++)
    printf("Turno: %d Aprovacoes: %d, Nota mais alta: %d\n",
           i, aprovacoes[i], alta[i]);
  return 0;
}
```

Exemplo 3: Número de Aprovações e Nota Mais Alta num Conjunto de Turnos

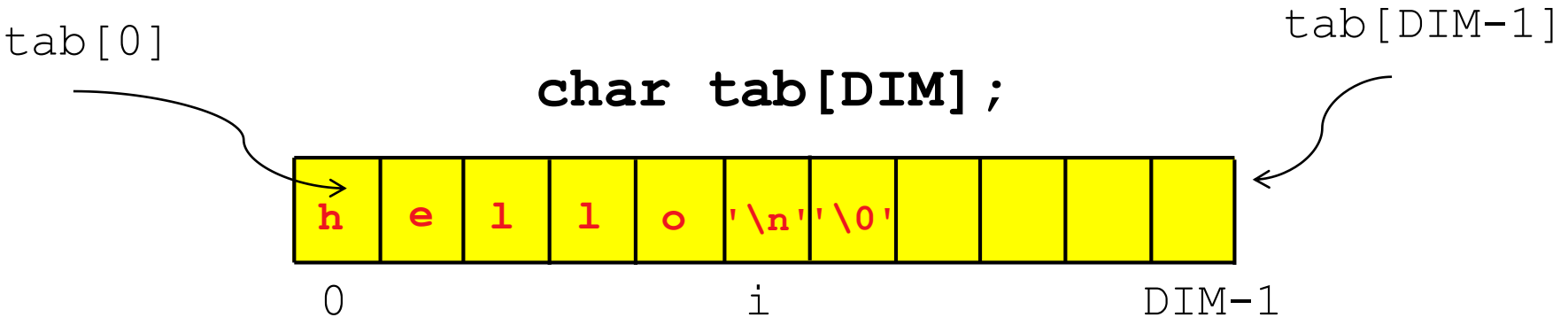
```
#define TURNOS 4
#define ALUNOS 25

int main () {
  int i, j, aprovacoes[TURNOS], alta[TURNOS], notas[TURNOS][ALUNOS];

  for (i = 0; i < TURNOS; i++)
    for (j = 0; j < ALUNOS; j++)
      scanf("%d", &notas[i][j]);
  ...
}
```

- Declaração de tabelas bidimensionais (matrizes)
- Variável `notas` é tabela de `4x25` inteiros: `TURNOS` é a 1ª dimensão (#linhas), `ALUNOS` é a 2ª dimensão (#colunas)
- `notas[i]` é o vector de inteiros no índice `i` da tabela `notas` (linha `i`); `notas[i][j]`é o inteiro na coluna `j` da linha `i`
- `notas[i][j]=0` atribui o valor `0` ao inteiro na coluna `j` da linha `i`

Strings (Tabelas de Caracteres)



- Convenção C: cadeia caracteres acaba com `'\0'`
- Cadeia de caracteres `"hello\n"` tem caracteres `'h'`, `'e'`, `'l'`, `'l'`, `'o'`, `'\n'` e `'\0'`
- `printf` espera strings neste formato

Exemplos 4 & 5: Leitura, Escrita e Cópia de Strings

- Objectivo:
  - Ler uma sequência de caracteres introduzidos pelo utilizador
  - Guardar os caracteres lidos numa "string"
  - Os caracteres `EOF` ou `'\n'` determinam o fim do conjunto de caracteres a ler
  - Mostrar a string lida
  - Efectuar a cópia da string lida para uma string do mesmo tamanho e mostrar as duas

Exemplo 4: Leitura e escrita de uma linha (I)

```
#include <stdio.h>

#define DIM 100

int main() {
  int c, i;
  char s[DIM];

  c = getchar();
  for (i = 0; i < DIM-1 && c != EOF && c != '\n'; i++) {
    s[i] = c;
    c = getchar();
  }

  s[i] = '\0';

  printf("%s\n", s);
  return 0;
}
```

Nós somos responsáveis por respeitar os limites!

Muito importante: Não me posso esquecer de adicionar o '\0' no fim!

Exemplo 4: Leitura e escrita de uma linha (II)

```c
#include <stdio.h>

#define DIM 100

int main() {
  int c, i;
  char s[DIM];

  for (i = 0; i<DIM-1 && (c =getchar()) != EOF && c != '\n'; i++)
    s[i] = c;

  s[i] = '\0';

  printf("%s\n", s);
  return 0;
}
```

Exemplo 4: Leitura e escrita de uma linha (III)

```c
#include <stdio.h>

#define DIM 100

int main() {
  int c, i = 0;
  char s[DIM];

  while (i < DIM-1 && (c = getchar()) != EOF && c != '\n')
      s[i++] = c;

  s[i] = '\0';

  printf("%s\n", s);
  return 0;
}
```

Exemplo 5: Cópia de Strings (I)

```
#include <stdio.h>
#define DIM 100

int main() {
  int c, i;
  char origem[DIM], destino[DIM];

  for (i = 0; i< DIM-1 && (c=getchar()) != EOF && c != '\n'; i++)
    origem[i] = c;

  origem[i] = '\0';

  for(i = 0; origem[i] != '\0'; i++)
    destino[i] = origem[i];
  destino[i] = '\0';

  printf("Origem: %s\nDestino: %s\n", origem, destino);
  return 0;
}
```

Exemplo 5: Cópia de Strings (II)

```
#include <stdio.h>
#define DIM 100

int main() {
  int c, i;
  char origem[DIM], destino[DIM];

  for (i = 0; i< DIM-1 && (c=getchar()) != EOF && c != '\n'; i++)
    origem[i] = c;

  origem[i] = '\0';

  i=0;
  while((destino[i] = origem[i]) != '\0')
    i++;

  printf("Origem: %s\nDestino: %s\n", origem, destino);
  return 0;
}
```

Exemplos 4 & 5: Leitura, Escrita e Cópia de Strings

- A Reter:
  - Strings são vectores de caracteres
  - Programador é responsável por respeitar os limites dos vectores
  - `'\0'` indica o fim da string
  - Formato de escrita (e leitura!) para strings `%s`
  - **Vectores são copiados posição a posição !**

Leitura de strings com o scanf (exemplo)

```
#include <stdio.h>

#define DIM 100

int main()
{
   char palavra[DIM];
   scanf("%s", palavra);

   printf("%s", palavra);

   return 0;
}
```

- O `scanf` permite ler uma string através da formatação %s
- A leitura é feita até encontrar um "*whitespace*" (' ', '`\n`', '`\t`', etc).
- O `scanf` introduz o '`\0`' no fim da leitura.
- Se quisermos ler uma linha inteira, teremos de usar o código dos slides anteriores (alternativa: comando **`fgets`**)

string.h

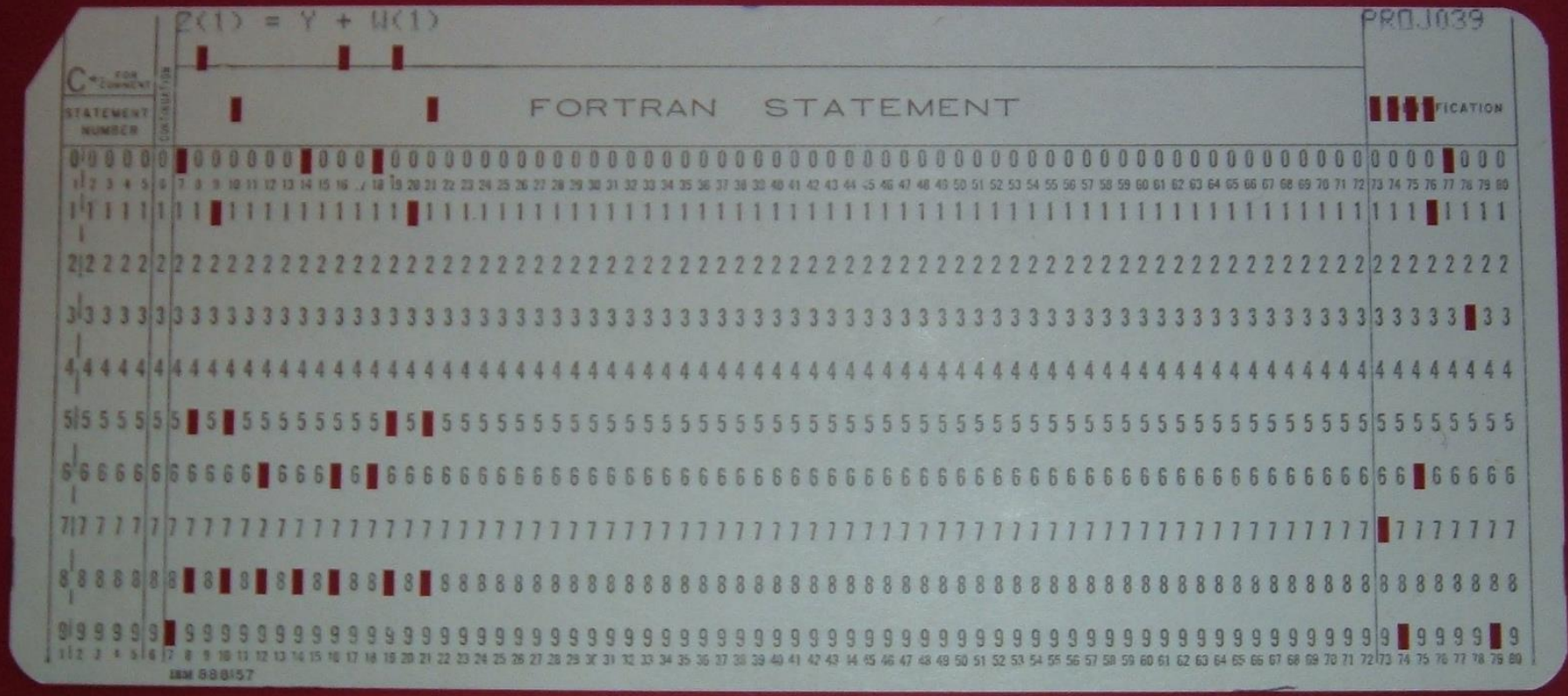
Ao fazer `#include <string.h>`

passamos a ter acesso a um conjunto razoável de funções para manipulação de strings. Exemplos úteis:

- `strcmp` (compara strings)
- `strcpy` (copia strings)
- `strdup` (duplica uma string)
- `strlen` (devolve o tamanho da string dada como argumento)
- etc.

Não se esqueçam do comando `man` (ex: `$ man strcpy`)

Cartão perfurado com uma linha de programa



Alguma Dúvida ?