



HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.



C PROGRAMMING LANGUAGE



ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

C PROGRAMMING LANGUAGE

WEEK 12: CHUỖI KÝ TỰ

ONE LOVE. ONE FUTURE.

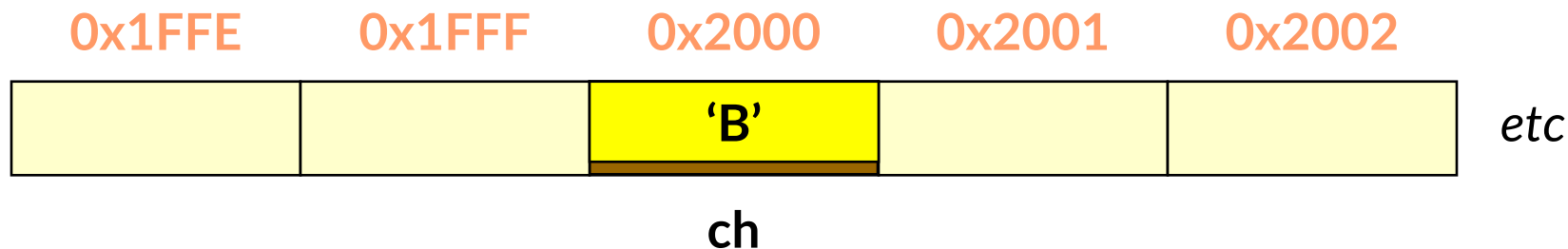
Biểu diễn trong bộ nhớ

- *Nhắc lại:*
 - Các ô trong bộ nhớ được đánh địa chỉ
 - Một khai báo biến cho phép dành một “ô” để chứa giá trị

Ví dụ:

```
char ch;
```

```
ch = 'B';
```

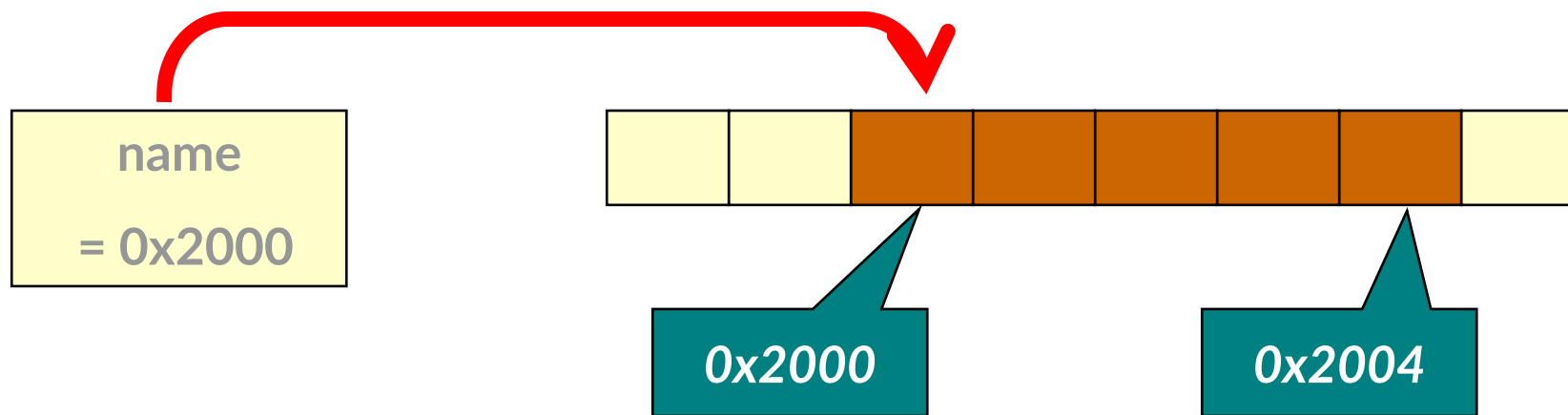


Biểu diễn chuỗi kí tự

- Một chuỗi được biểu diễn bởi một mảng kí tự
- Mỗi phần tử của mảng chứa một `char`
- Tên chuỗi là địa chỉ trỏ đến phần tử đầu tiên của mảng

Ví dụ:

```
char name[5];
```



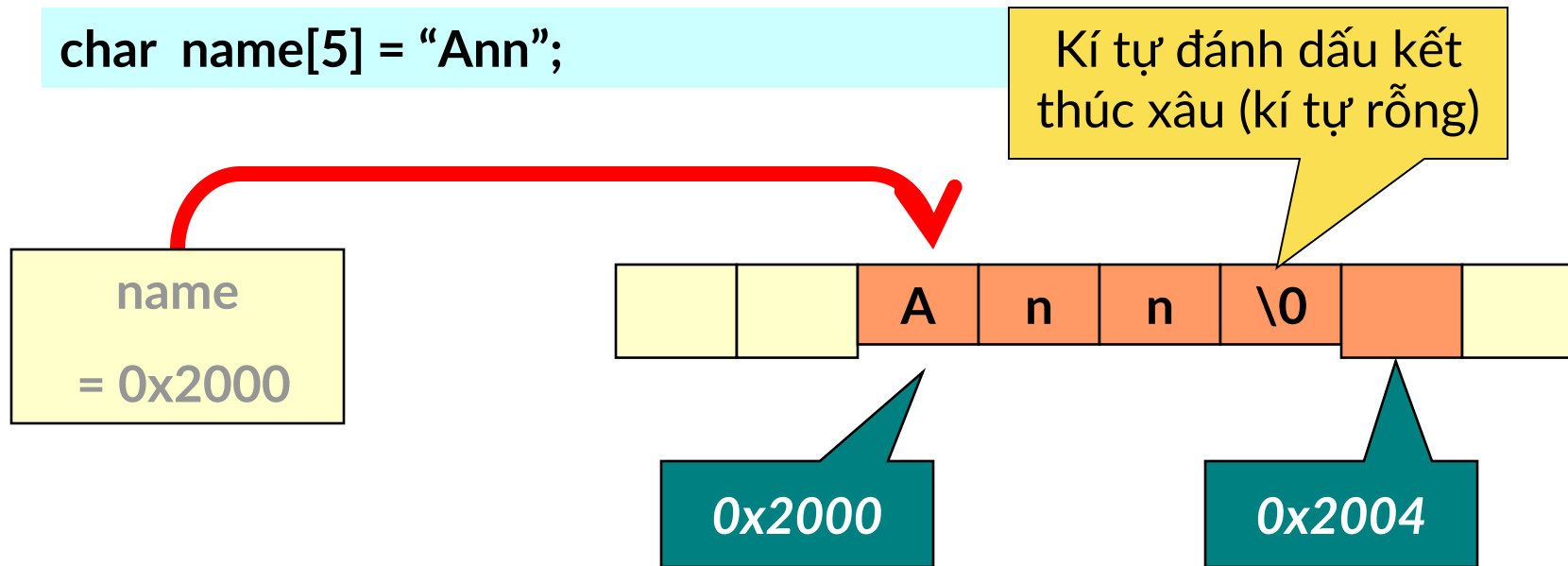
Mảng kí tự vs. Chuỗi kí tự

- Một chuỗi kí tự là một mảng kí tự nhưng không phải là điều ngược lại
- Một chuỗi kí tự phải có **kí tự kết thúc** hay còn gọi là kí tự rỗng (`'\0'`)
- Kí tự kết thúc dùng để báo hiệu điểm dừng của xâu. Nó tiện cho việc xử lí xâu trong các hàm, ví dụ như hàm `printf()`, `scanf()`, v.v.

Khai báo chuỗi

Khai báo 1:

```
char name[5] = "Ann";
```



Khai báo tương đương:

```
char name[5] = {'A','n','n','\0'};
```

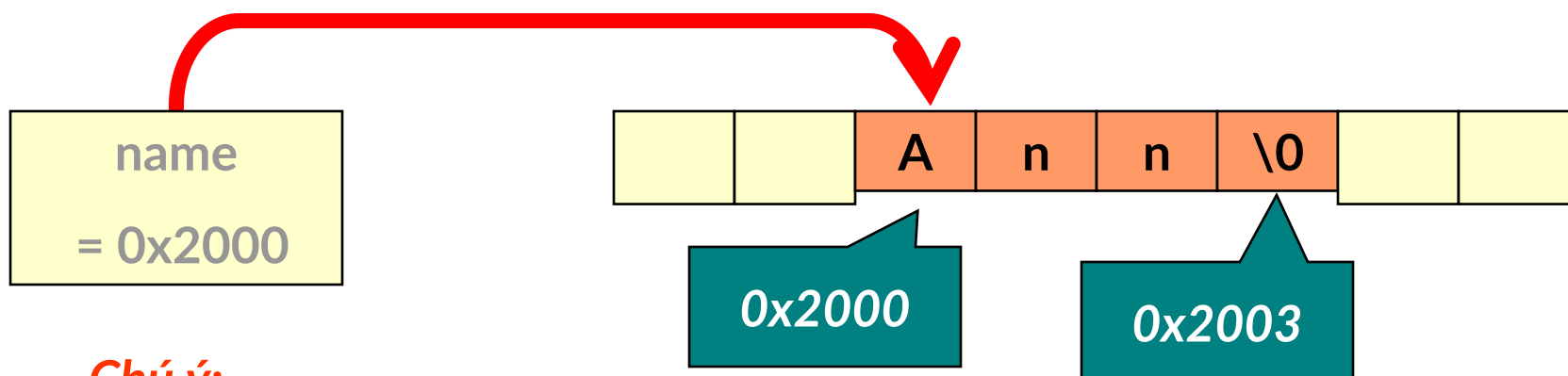
Khai báo chuỗi (tiếp)

Khai báo 2:



```
char name[] = "Ann";
```

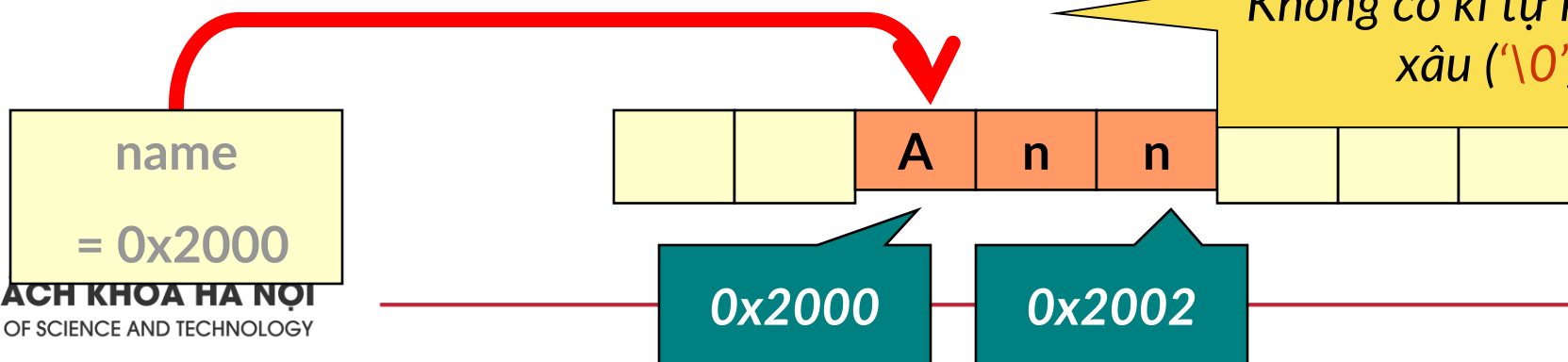
Lấy thêm một ô kí tự
cho '\0'



Chú ý:

```
char name[] = {'A','n','n'};
```

Không có kí tự kết thúc
xâu ('\0')

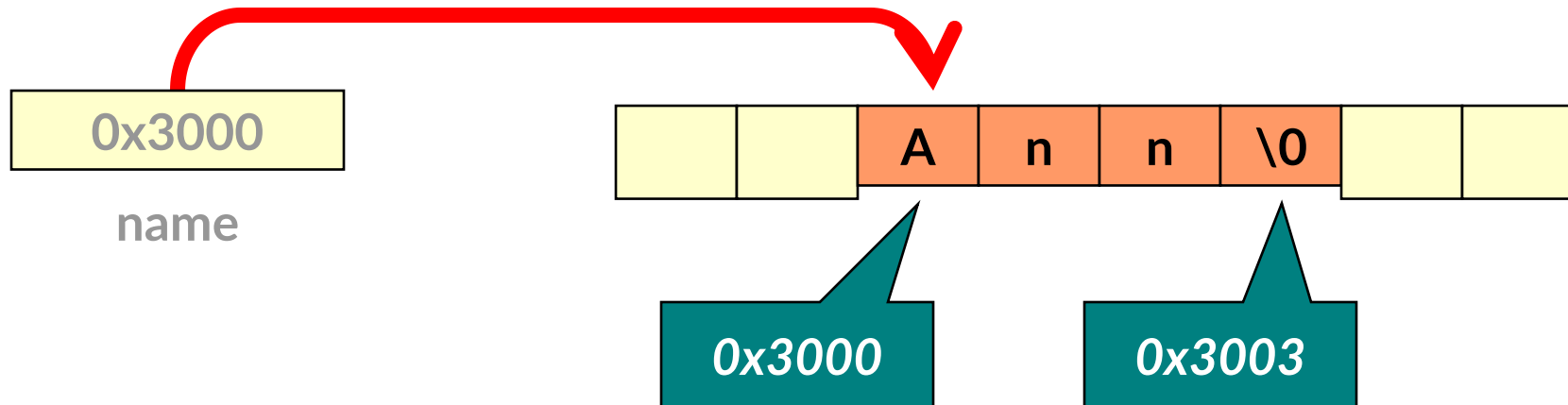


Khai báo chuỗi (tiếp)

Khai báo 3:



```
char *name = "Ann";
```



Xuất nhập chuỗi

```
#include <stdio.h>
```

```
#define MAXLENGTH 15
```

```
int main()  
{
```

```
    char str1[MAXLENGTH];  
    char str2[MAXLENGTH];
```

```
    scanf("%s", str1);  
    gets(str2);  
    printf("%s\n%s\n", str1, str2);
```

```
    return 0;
```

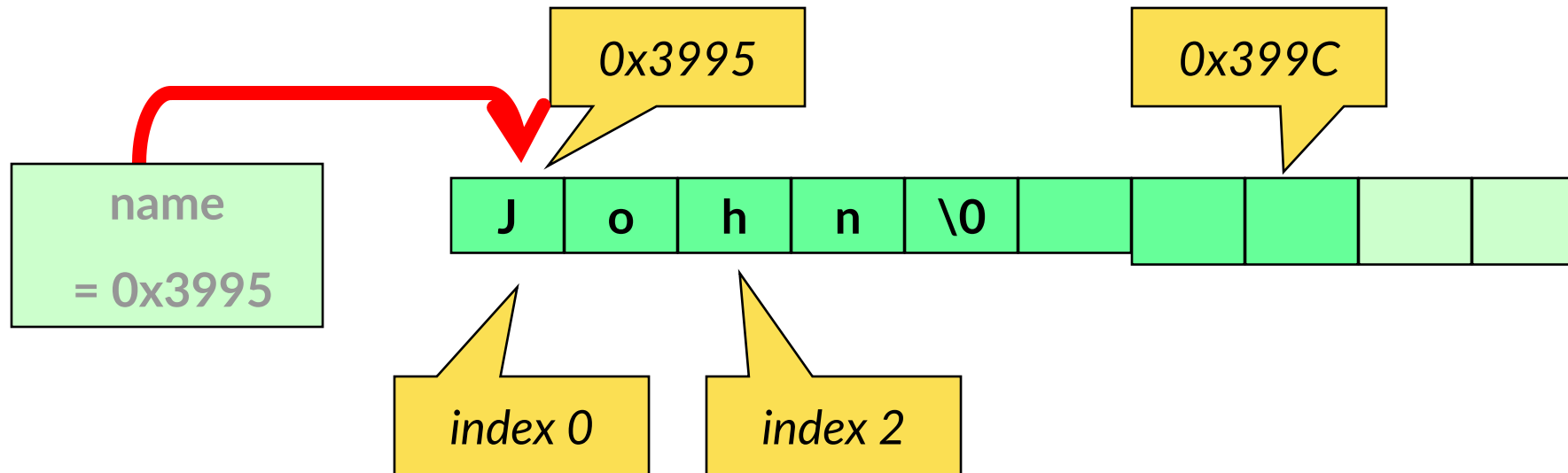
Đặt tên cho
hằng số

Số kí tự tối đa có
trong chuỗi là
MAXLENGTH-1

Không có
toán tử **&** ở
đây

gets() cho phép
nhập chuỗi với kí tự
trắng

Kí tự trong chuỗi



```
char name[8] = "John";  
int i = 2;  
  
printf("Char at index %d is %c.\n", i, name[i]);
```

output: Char at index 2 is h.

Chương trình đếm kí tự

- Đếm số kí tự không phải là kí tự trắng trong một chuỗi được nhập vào

```
#include <stdio.h>

int main()
{
    char str[80];
    int dem, i;

    printf("Nhap xau bat ki: ");
    gets(str);

    dem = 0; i = 0;
    while ( str[i] != '\0' ) {
        if ( str[i] != ' ' ) dem++;
        i++;
    }
    printf("So ki tu khac trang trong xau la %d", dem);

    return 0;
}
```

Xử lý chuỗi

- `#include <string.h>`
- Phải sử dụng các hàm thư viện cho các thao tác xử lý chuỗi:
 - Gán: `strcpy()`
 - Nối chuỗi: `strcat()`
 - So sánh: `strcmp()`
 - Lấy độ dài xâu: `strlen()`
 - v.v.

```
#include <stdio.h>
#include <string.h>

#define MAXLENGTH 100

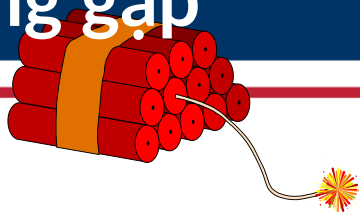
int main()
{
    char string1[MAXLENGTH];
    char string2[MAXLENGTH];

    strcpy(string1, "Hello World!");
    strcpy(string2, string1);

    return 0;
}
```

string1: "Hello World!"
string2: "Hello World!"

Lỗi thường gặp



Dùng toán tử gán không tương thích kiểu

```
char name1[5] = "Ann";  
char name2[5] = "Dave";  
name2 = name1;
```



Không đủ bộ nhớ

```
char name[] = "Ann";  
strcpy(name, "David");
```



name

A	n	n	\0						
---	---	---	----	--	--	--	--	--	--

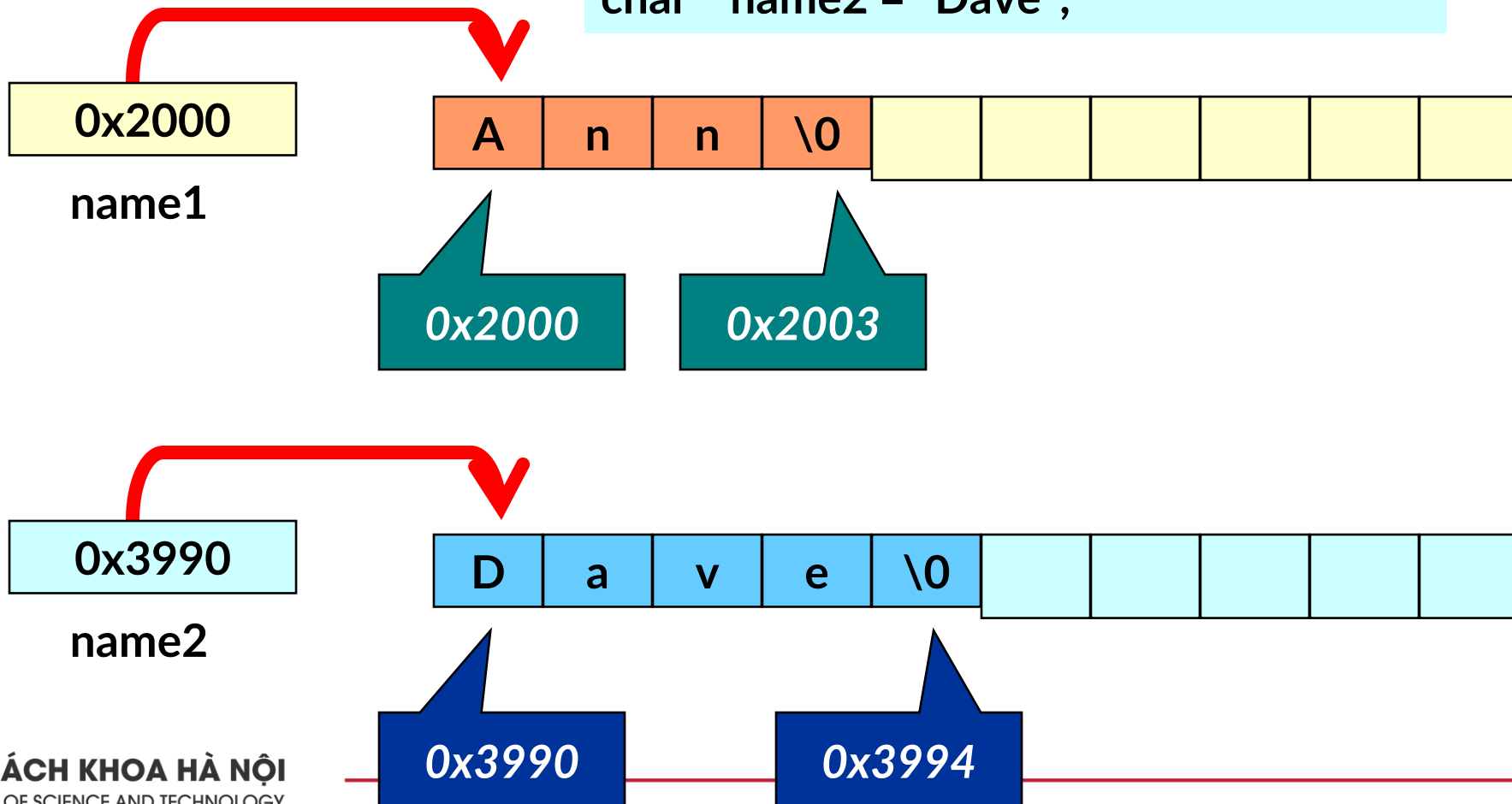
name

D	a	v	i	d	\0				
---	---	---	---	---	----	--	--	--	--

Gán dạng con trỏ



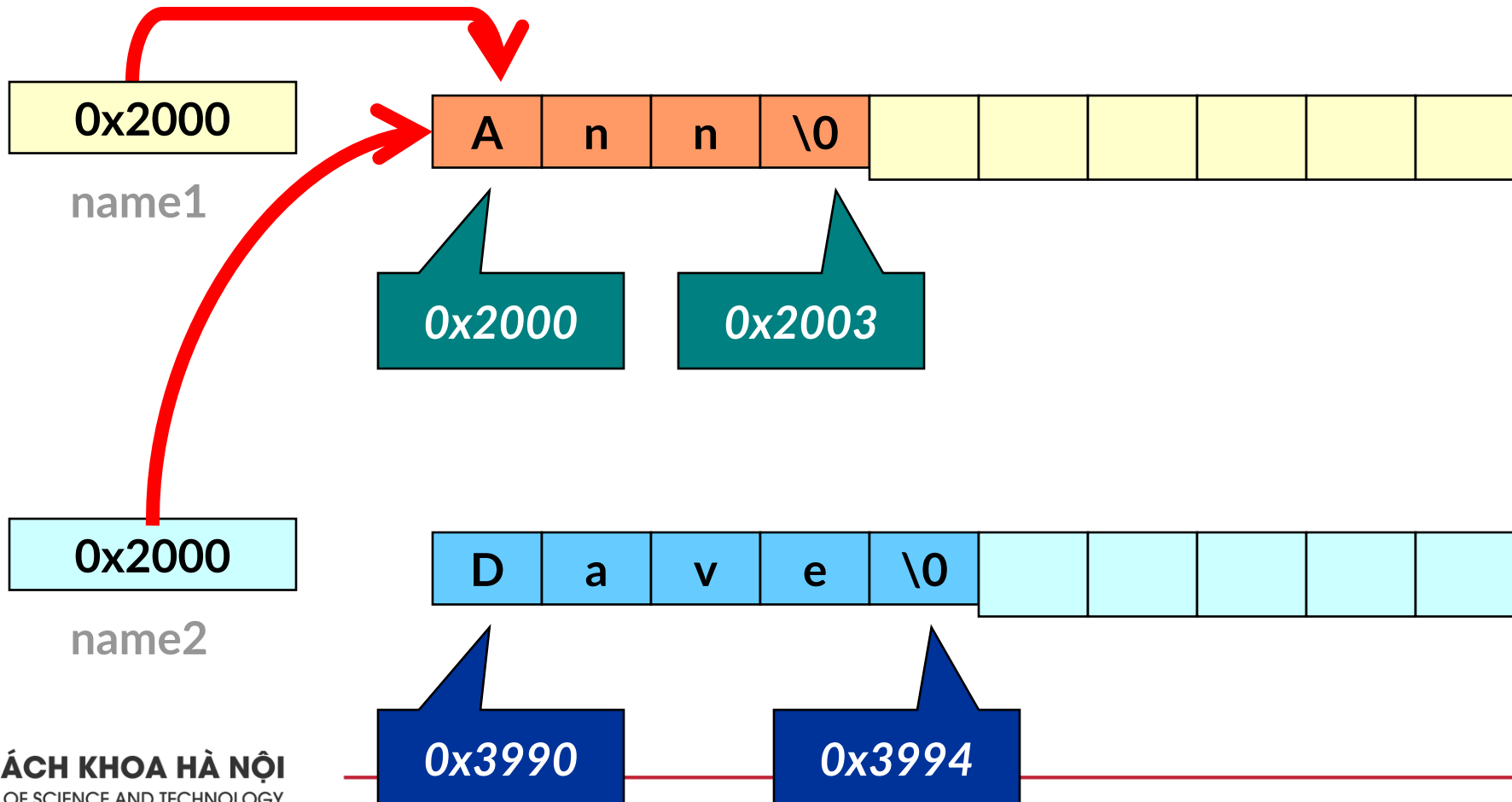
```
char *name1 = "Ann";  
char *name2 = "Dave";
```



Gán dạng con trỏ (tiếp)



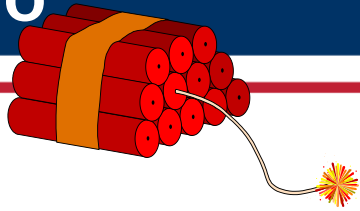
name2 = name1;



```
char string1[80];  
char string2[80];  
  
strcpy(string1, "Goodbye");  
strcpy(string2, ", Cruel ");  
  
strcat(string1, string2);  
strcat(string1, string2);  
strcat(string1, "World!");
```

string1: "Goodbye, Cruel , Cruel World!"
string2: ", Cruel "

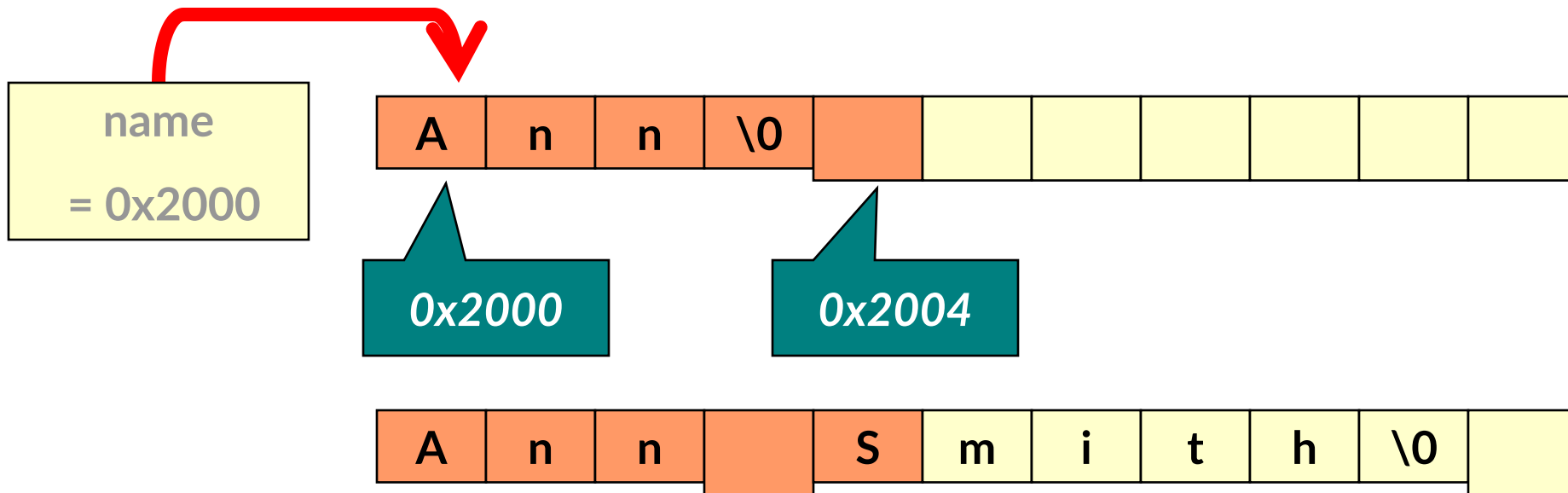
Lỗi bộ nhớ



```
char name[5];
```

```
strcpy(name, "Ann");
```

```
strcat(name, " Smith");
```



So sánh chuỗi

```
strcpy(string1, "Apple");  
strcpy(string2, "Wax");
```

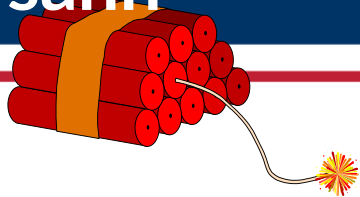
```
if (strcmp(string1, string2) < 0)  
{  
    printf("%s %s\n", string1, string2);  
}  
else  
{  
    printf("%s %s\n", string2, string1);  
}
```

Trả về

0 : nếu string1 giống string2
< 0: nếu string1 < string2
> 0: nếu string1 > string2

output: Apple Wax

Lỗi về so sánh



```
strcpy(string1, "Apple");  
strcpy(string2, "Wax");
```

Chỉ là phép so
sánh địa chỉ bộ
nhớ



```
if (string1 < string2)  
{  
    printf("%s %s\n", string1, string2);  
}  
else  
{  
    printf("%s %s\n", string2, string1);  
}
```

Tham số hàm là chuỗi

- Giống như mảng ta có thể khai báo chuỗi như là tham số với **char*** hoặc **char[]**
`void greeting (char* name)`
`void greeting (char name[])`
- Chúng trỏ đến kí tự đầu tiên của chuỗi (mảng kí tự)
- Thay đổi nội dung chuỗi truyền vào sẽ làm thay đổi chuỗi kí tự gốc
- Không cần truyền số kí tự trong chuỗi cho hàm

Viết hàm chuyển đổi chữ thường thành chữ hoa

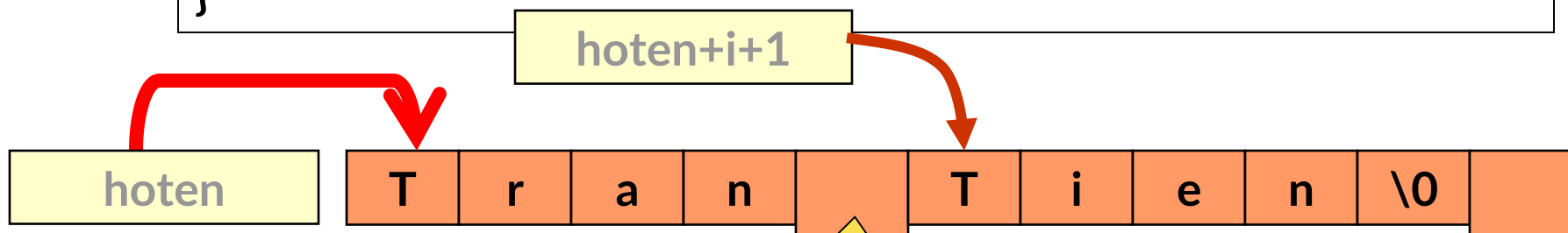
```
char *capitalize(char * str)
{
    for (i=0; i<strlen(str); i++)
        if ( str[i]>='a'&& str[i]<='z')&&
            (i==0||str[i-1]==' ')
            str[i] = 'A' + (str[i]-'a');
    return str;
}
```

Chương trình tách tên

- Viết một hàm trả về con trỏ tới phần tên của một xâu họ và tên là tham số của hàm

```
char * timten(const char[] hoten)
{
    int i;
    i = strlen(hoten)-1;
    /* tìm kí tự trắng cuối cùng trong xâu */
    while (i >= 0 && hoten[i] != ' ') i--;
    return hoten + i + 1;
}
```

Giả thiết họ tên là xâu không rỗng và không có kí tự trắng thừa



Vị trí kí tự trắng được tìm thấy

(i=4)

Chương trình tách tên (tiếp)

```
#include <stdio.h>
#include <string.h>

char * timten(const char[] hoten)

int main()
{
    char hoten[80];

    printf("Nhap mot xau ho va ten: ");
    gets(hoten);

    printf("Ten sau khi tach duoc: %s", timten(hoten));

    return 0;
}
```

Khai báo **const** thể hiện giá trị của tham số không bị thay đổi trong hàm

Bài tập

- Viết chương trình với các hàm xử lý chuỗi được tạo ra để làm các việc sau:
 - Cắt bỏ các dấu cách thừa trong một chuỗi
 - Đếm số từ hiện có trong một chuỗi
 - Đảo ngược chuỗi
 - Tách một chuỗi họ và tên thành hai chuỗi mới cho phần họ và phần tên



A large graphic on the left side of the slide. It consists of a dark blue background with a circular pattern of red dots of varying sizes, creating a sense of depth and movement. The word "HUST" is centered within this graphic in a white, bold, sans-serif font.

HUST

THANK YOU !