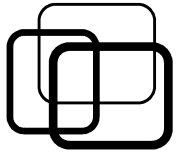
Tập tin nhị phân

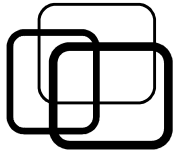
GV. Nguyễn Minh Huy

Nội dung



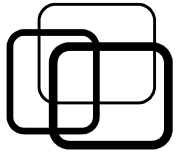
- Chuỗi cấp phát động.
- Tập tin nhị phân.
- Tham số hàm main.

Nội dung



- **Chuỗi cấp phát động.**
- Tập tin nhị phân.
- Tham số hàm main.

Chuỗi cấp phát động



- Con trỏ ký tự:
  - Chuỗi = mảng ký tự + ký tự '\0';

char **s1[6]** = { 'H', 'e', 'l', 'l', 'o', '\0' }; **s1**

H	e	l	l	o	\0

char **s2[ ]** = "Hello"; **s2**

H	e	l	l	o	\0

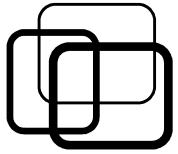
  - Chuỗi cấp phát động:
    - Mảng ký tự động → phải cấp phát và thu hồi vùng nhớ.
    - Có thể thay đổi kích thước khi cần.
    - Khai báo: char ***<chuỗi>**;

char ***s3** = new char[6]; **s3** 27 → | ? | ? | ? | ? | ? | ? |

//…

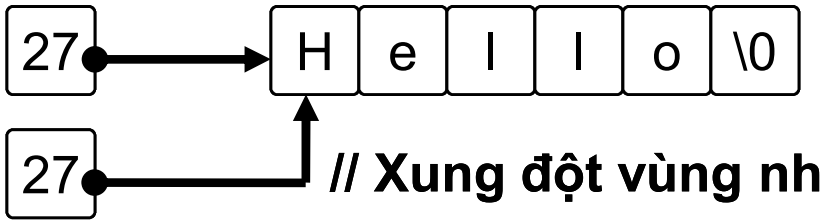
delete [ ]s3;

Chuỗi cấp phát động



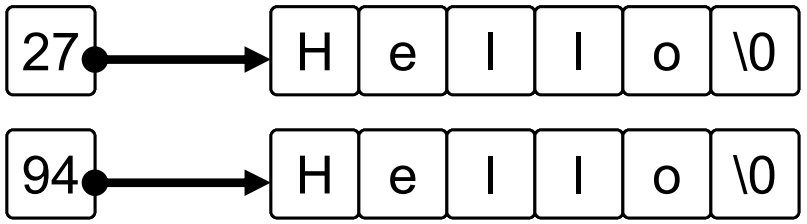
- Thư viện <string.h>:
  - Lệnh sao chép chuỗi:
    - Không sao chép bằng toán tử =.

```
char  *s1 = “Hello”;
char  *s2 = s1;
```

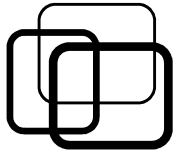


    - Thao tác sao chép chuỗi:
      - B1: khai báo và cấp phát chuỗi mới.
      - B2: dùng strcpy để sao chép.

```
char  *s1 = “Hello”;
char  *s2 = new char[ strlen(s1) + 1 ];
strcpy( s2, s1 );
```



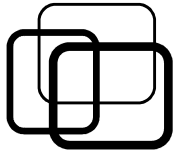
Chuỗi cấp phát động



- Thư viện <string.h>:
  - Lệnh tạo bản sao:
    - Cú pháp: **strdup**(<chuỗi nguồn>);
    - Trả về: chuỗi bản sao, vùng nhớ được tự động cấp phát.
    - Phải thu hồi vùng nhớ chuỗi bản sao khi dùng xong.

```
char  s1 = “Hello”;
char  *s2 = strdup(s1);

// Tương tự….
// char *s2 = new char[ strlen(s1) + 1 ];
// strcpy( s2, s1 );

free(s2);
```

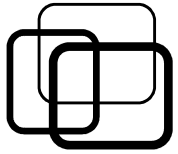
Chuỗi cấp phát động



- Thư viện <string.h>:
  - Lệnh so sánh chuỗi:
    - Cú pháp: **strcmp**(<chuỗi 1>, <chuỗi 2>);
    - Trả về: 0 (bằng), 1 (lớn hơn), -1 (nhỏ hơn).
    - So sánh nội dung 2 chuỗi theo thứ tự từ điển.

```
char  *s1 = “abc”;
char  *s2 = “abaab;
char  s3[10];
strcpy(s3, s1);

int     kq1 = strcmp(s1, s2);   // kq1 = 1.
int     kq2 = strcmp(s1, s3);   // kq2 = 0.
int     kq3 = strcmp(s2, s3);   // kq2 = -1.
```

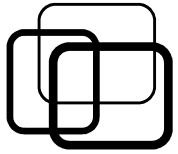
Chuỗi cấp phát động



- Thư viện <string.h>:
  - Lệnh nối chuỗi:
    - Cú pháp: **strcat**(<chuỗi đích>, <chuỗi nguồn>);
    - Nối chuỗi nguồn và cuối chuỗi đích.
    - Chuỗi đích phải đủ vùng nhớ để nối!!

```
char  *s1 = “Hello”;
char  *s2 = “World;
char  *s3 = new char[ strlen(s1) + strlen(s2) + 1 ];

strcat(s3, s1);          // Nối s1 vào s3.
strcat(s3, s2);          // Nối tiếp s2 vào s3.
```

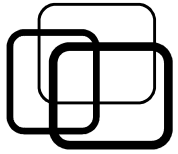
Chuỗi cấp phát động



- Thư viện <string.h>:
  - Lệnh tìm chuỗi con:
    - Cú pháp: **strstr**(<chuỗi nguồn>, <chuỗi con>);
    - Trả về: địa chỉ vị trí đầu tiên xuất hiện chuỗi con (tìm thấy), NULL (thất bại).

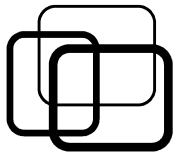
```
char  s1[ ] = “Hello World”;
char  *s2 = “World;
char  *s3 = strstr(s1, s2);

if (s3 == NULL)
    printf(“Không tìm thấy chuỗi con.”);
else
    printf(“Vị trí xuất hiện chuỗi con = %d”, s3 - s1);
```

Chuỗi cấp phát động



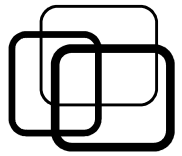
- Bài tập:
  - Nhập chuỗi động:
    - Khai báo kiểu SinhVien:
      - Mã số: cố định 8 ký tự.
      - Họ tên: tối đa 50 ký tự.
      - Điểm trung bình: số thực.
    - Viết hàm nhập thông tin 1 sinh viên.

Nội dung



- Chuỗi cấp phát động.
- **Tập tin nhị phân.**
- Tham số hàm main.

Tập tin nhị phân



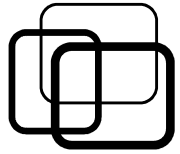
- Thiết bị nhập xuất:
  - Chương trình là "cỗ máy" làm việc.

    ➔ Dữ liệu → Chương trình → Kết quả.
  - Chương trình lấy dữ liệu và xuất kết quả thế nào?

    ➔ Thiết bị nhập xuất (IO Device).
  - Phân loại thiết bị:
    - Thiết bị nhập: bàn phím, con chuột, tập tin, …
    - Thiết bị xuất: màn hình, máy in, tập tin, …

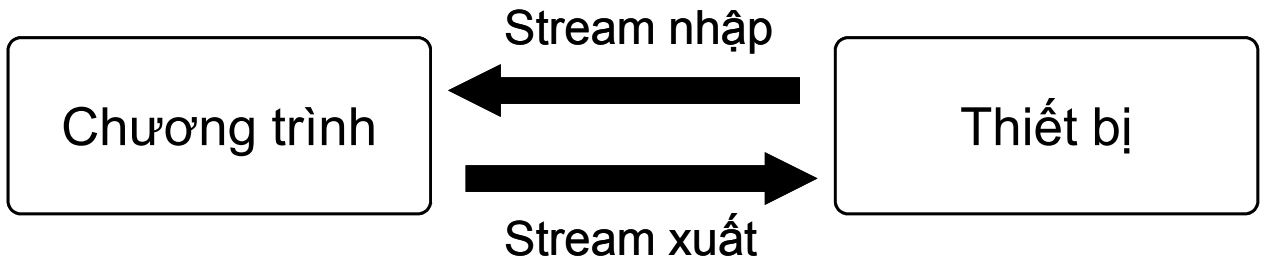
    ➔ Tập tin là thiết bị vừa nhập vừa xuất.

Tập tin nhị phân

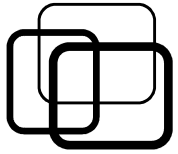


- Khái niệm stream:
  - Chương trình giao tiếp với thiết bị bằng cách nào?

  ➔ Thông qua "dòng chảy" dữ liệu.
  - Stream: "dòng chảy" kết nối chương trình và thiết bị.
  - Phân loại stream:
    - Stream nhập: "dòng chảy" kết nối thiết bị nhập.
    - Stream xuất: "dòng chảy" kết nối thiết bị xuất.

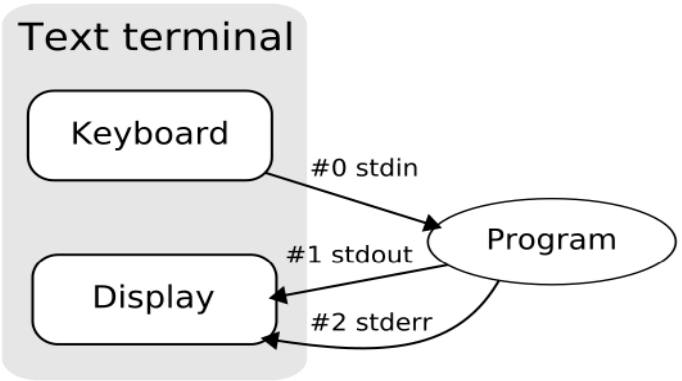


Tập tin nhị phân



- Stream trong C:
  - Các stream khai báo sẵn trong C:

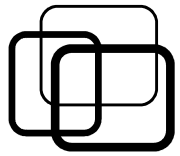
Stream	Ý nghĩa	Thiết bị kết nối
stdin	Stream nhập chuẩn.	Bàn phím
stdout	Stream xuất chuẩn.	Màn hình
stderr	Stream lỗi chuẩn.	Màn hình



  - Lệnh nhập xuất tổng quát:
    - **fscanf(**<Stream>, "<Định dạng kiểu>", &<Biến 1>, …**)**;
    - **fprintf(**<Stream>, "<Định dạng xuất>", <Biến 1>, …**)**;

```
fscanf( stdin, "%d", &x);          // Nhập từ bàn phím.
fprintf( stdout, "Hello World" );  // Xuất ra màn hình.
```

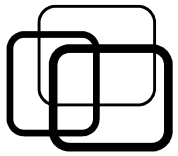
Tập tin nhị phân



- Stream trong C++:
  - Nâng cấp stream của C:
    - Tương thích với stream của C.
    - Sử dụng tiếp cận hướng đối tượng.
    - Dễ sử dụng hơn.
  - Các stream khai báo sẵn trong C++:

Stream	Ý nghĩa	Thiết bị kết nối
cin	Stream nhập chuẩn.	Bàn phím
cout	Stream xuất chuẩn.	Màn hình
cerr	Stream lỗi chuẩn.	Màn hình

Tập tin nhị phân



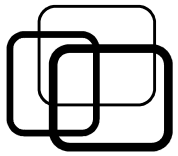
- Stream trong C++:
  - Toán tử nhập >>:
    - Nhập dữ liệu từ stream.
    - Không cần định dạng kiểu.
    - Cú pháp:

      <stream> >> <biến>;
  - Toán tử xuất <<:
    - Xuất dữ liệu ra stream.
    - Không cần định dạng kiểu.
    - Cú pháp:

      <stream> << <biến/trị>;

```
int main()
{
    int     n;
    float  a[10];

    cout << "Nhap n = ";
    cin >> n;

    for (int i = 0; i < n; i++)
    {
       cout << "Nhap a[" << i << "] = ";
       cin >> a[ i ];
    }
}
```

Tập tin nhị phân

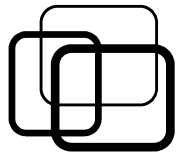


- Tập tin vs. Bộ nhớ:

Tiêu chí	Tập tin (Bộ nhớ ngoài)	Bộ nhớ
Tốc độ xử lý	Chậm	Nhanh
Khả năng truy xuất	Tuần tự	Ngẫu nhiên
Chi phí	Rẻ tiền	Đắc tiền
Dung lượng	Lớn	Nhỏ
Thời gian	Lưu trữ lâu dài	Lưu trữ nhất thời

- Tập tin vs. Bàn phím và Màn hình:
  - Không cần thông qua người dùng.
  - Đọc/ghi tự động, nhiều lần.
  - Giao tiếp được với chương trình khác.

Tập tin nhị phân

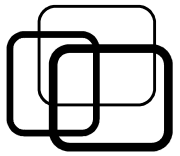


- File stream:
  - "Dòng chảy" kết nối chương trình và tập tin.
  - Khái báo: **FILE** *<tên stream>;

    ➔ Con trỏ tập tin.

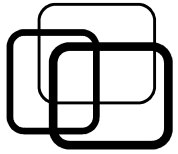
    FILE *f1;
  - Các bước xử lý tập tin:
    - Bước 1: mở tập tin.
    - Bước 2: thao tác trên tập tin.
    - Bước 3: đóng tập tin.

Tập tin nhị phân



- Các lệnh thao tác tập tin văn bản:
  - Mở tập tin: fopen, freopen.
  - Đóng tập tin: fclose.
  - Đọc/ghi tập tin: fscanf, fgets, fprintf.
- Bài tập:
  - Viết hàm đếm độ dài dòng n trong tập tin văn bản.
    - int getLineLength(<đường dẫn tập tin>, n).
    - Dòng đầu tiên: n = 0.
    - Hàm trả về -1 nếu lỗi.

Tập tin nhị phân

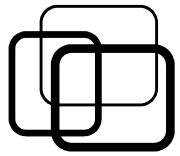


- Chế độ mở nhị phân:
  - Bảng chế độ mở tập tin:

Chế độ mở	Ý nghĩa
r	**R**ead-only, mở để đọc dữ liệu (kiểu text).
w	**W**rite-only, mở để ghi dữ liệu (kiểu text).
a	**A**ppend-only, mở để ghi thêm dữ liệu (kiểu text).
[Chế độ mở]+	Kết hợp đọc, ghi cùng lúc.
[Chế độ mở]b	Mở kiểu nhị phân (binary).

  - Cho phép đọc/ghi thô từng byte trên tập tin.
  - Ánh xạ từng byte tập tin vào từng byte bộ nhớ.

Tập tin nhị phân

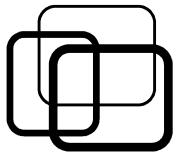


- Lệnh fread:
  - Đọc block bytes từ tập tin.
    - Cú pháp: **fread(**<Địa chỉ vùng nhớ>, <Kích thước block> , <Số block cần đọc>, <Con trỏ tập tin>**)**;
    - Trả về: số block đọc được.

    ➔ Kết thúc tập tin khi số block đọc được < số block cần đọc.

```
int     x;
char   *p = new char[ 100 ];
FILE  *f = fopen("C:\\BaiTap.txt", "rb");

if ( f != NULL )
{
    fread( &x, sizeof(int), 1, f );     // Đọc 4 bytes vào số nguyên x.
    fread( p, sizeof(char), 100, f );// Đọc 100 bytes vào vùng nhớ p.
    fclose( f );
}
```

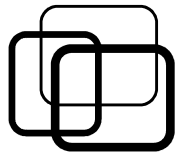
Tập tin nhị phân



- Lệnh fwrite:
  - Ghi block bytes vào tập tin.
    - Cú pháp: **fwrite(**<Địa chỉ vùng nhớ>, <Kích thước block> , <Số block cần đọc>, <Con trỏ tập tin>**)**;
    - Trả về: số block ghi được.

```
int      x = 123456;
char   s[ ] = “Hello World”;
FILE  *f = fopen(“C:\\BaiTap.txt”, “wb”);

if ( f != NULL )
{
     fwrite( &x, sizeof(int), 1, f );            // Ghi 4 bytes số nguyên x.
     fwrite( s, sizeof(char), strlen(s), f );  // Ghi 11 bytes chuỗi s.
     fclose( f );
}
```

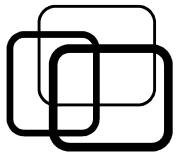
Tập tin nhị phân



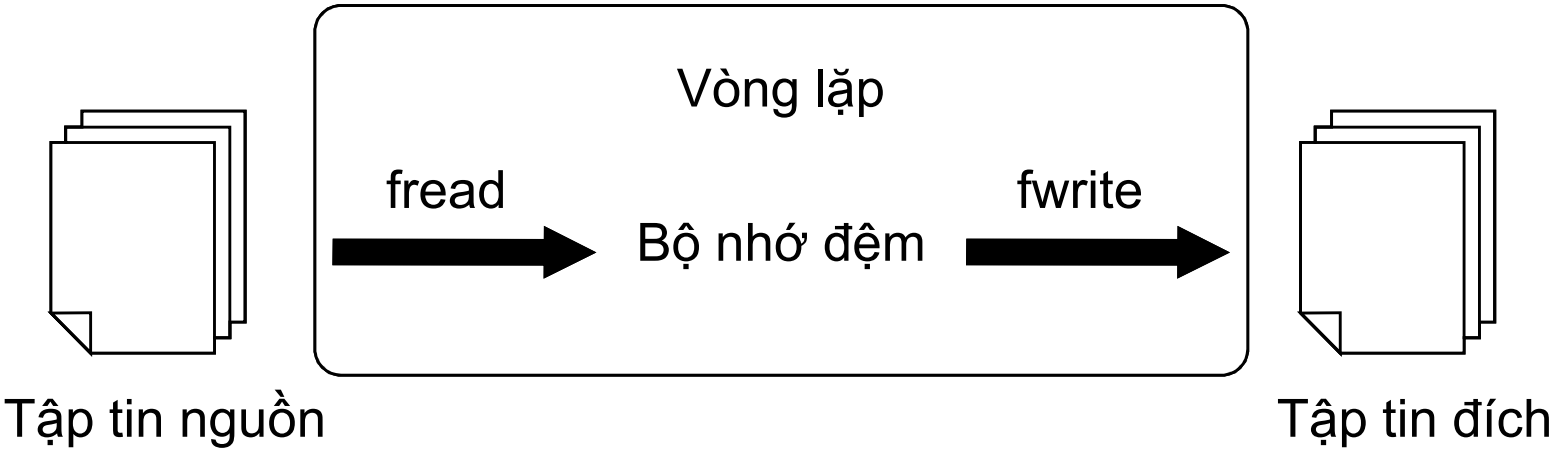
- Lệnh fseek:
  - Thay đổi vị trí con trỏ tập tin.
    - Cú pháp: **fseek(**<Con trỏ tập tin>, <Độ dời>, <Vị trí gốc>**)**;
    - <Vị trí gốc>:
      - SEEK_SET (đầu tập tin).
      - SEEK_CUR (vị trí hiện hành).
      - SEEK_END (cuối tập tin).
    - Chỉ làm việc với tập tin đang mở.

```
FILE  *f = fopen(“C:\\BaiTap.txt”, “r”);
if ( f != NULL )
{
     fseek( f, 2, SEEK_CUR );  // Chuyển con trỏ tới 2 bytes.
     fclose( f );
}
```

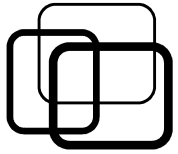
Tập tin nhị phân



- Bài tập:
  - Viết hàm sao chép tập tin:
    - bool copyFile(<đường dẫn nguồn>, <đường dẫn đích).

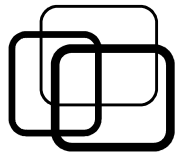


Nội dung



- Chuỗi cấp phát động.
- Tập tin nhị phân.
- **Tham số hàm main.**

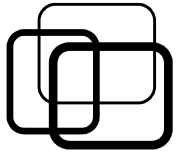
Tham số hàm main



- Khái niệm tham số dòng lệnh:
  - Chương trình là một hàm khổng lồ!!
  - Truyền dữ liệu đầu vào cho chương trình thế nào?
  - Tham số dòng lệnh:
    - Tham số truyền khi gọi thực hiện chương trình.
    - Hàm main có thể nhận các tham số này để xử lý.
  - Cách truyền tham số:
    - Gọi chương trình ở chế độ command line của hệ điều hành.
    - Cú pháp: <chương trình> **<tham số 1> <tham số 2> …**

      C:\>BaiTap\baitap1.exe **hello 5 /abc**
      C:\>copy **C:\BaiTap\baitap1.exe D:\Files\baitap1.exe**

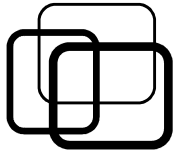
Tham số hàm main



- Sử dụng tham số hàm main:
  - Khai báo nhận tham số:
    - Cú pháp: int **main**(int **argc**, char ****argv**);
    - argc: số lượng tham số.
    - argv: danh sách các tham số.
    - Các tham số ở dạng chuỗi ký tự.
    - Tham số đầu tiên là tên chương trình.

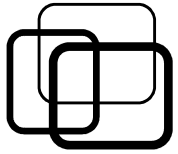
```
int main(int  argc, char  **argv)
{
        cout << “So luong tham so = “ << argc;
        cout << “Danh sach tham so:” << endl;
        for (int i = 0; i < argc; i++)
              cout << argv[ i ] << endl;
}
```

Tóm tắt



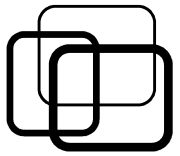
- Stream nhập xuất:
  - "Dòng chảy" kết nối chương trình và thiết bị.
  - Nhập xuất trong C: printf, scanf.
  - Nhập xuất trong C++: cin >>, cout <<.
  - File stream: con trỏ tập tin, fopen, fprintf, fscanf.
- Chế độ mở nhị phân:
  - Đọc/ghi thô từng byte.
  - Lệnh thao tác: fread, fwrite, fseek.

Tóm tắt



- Chuỗi cấp phát động:
  - Dùng mảng động để biểu diễn chuỗi động.
  - Không sao chép chuỗi bằng toán tử =.
  - Thư viện <string.h>: strcpy, strdup, strcmp, strstr.
- Tham số hàm main:
  - Tham số dòng lệnh truyền vào chương trình.
  - Cú pháp: int main(int argc, char **argv);

Bài tập



Bài tập 4.1:

Viết chương trình C/C++ thực hiện những việc sau:

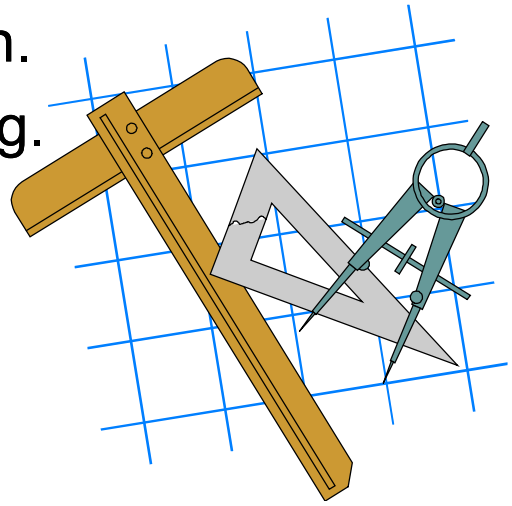
- Nhập vào một đoạn văn dài đến khi kết thúc bằng dấu '.' và xuống hàng.
- Hãy xuất ra những thông tin sau:

a) Số từ trong đoạn văn (các từ cách nhau khoảng trắng hoặc dấu câu ',' '.' ';' '?', '!').

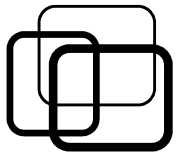
b) Đoạn văn đã chuẩn hóa

+ Bỏ khoảng trắng đầu và cuối đoạn văn.
+ Các từ cách nhau đúng 1 khoảng trắng.
+ Viết hoa chữ cái đầu mỗi từ.

Yêu cầu: dùng chuỗi cấp phát động.



Bài tập



Bài tập 4.2:

Viết chương trình C/C++ sao chép tập tin bằng tham số dòng lệnh.

Cú pháp dòng lệnh chương trình COPY:

- Cú pháp 1: sao chép đặt tên mới:

  COPY **<file nguồn> <file đích>**

- Cú pháp 2: sao chép giữ tên cũ:

  COPY **<file nguồn> <folder đích>/**

- Cú pháp 3: nối tập tin:

  COPY **<file 1> + <file 2> <file đích>**

- Cú pháp 4: xem hướng dẫn:

  COPY **-?**