

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN – TRUYỀN THÔNG



BÁO CÁO BÀI TẬP THỰC HÀNH NHÓM
Môn nguyên lý hệ điều hành

Hà Nội, 04-2021

GIỚI THIỆU

Trong bản báo cáo này, nhóm em sẽ trình bày về chương trình Shell, TinyShell, là chương trình C/C++ được biên dịch trên hệ điều hành Windows.

Dù đã rất cố gắng nhưng do khả năng còn hạn chế nên không thể không có sai sót, rất mong được thầy tận tình chỉ bảo.

Xin chân thành cảm ơn!

Nhóm sinh viên: Trịnh Quang Hùng – 20190054 – CTTN_KHMT_K64

Trần Tiến Bằng – 20193988 – CTTN_KHMT_K64

Trịnh Hồng Phụng – 20190062 – CTTN_KHMT_K64

Mục lục

I.	Giới thiệu chương trình	1
I.1.	<i>TinyShell</i>	1
I.2.	<i>Hướng dẫn cài đặt</i>	1
II.	Background/Foreground	2
III.	Các câu lệnh quản lý tiến trình	2
III.1.	<i>Liệt kê các tiến trình nền</i>	2
III.2.	<i>Kết thúc một tiến trình</i>	3
III.3.	<i>Tạm dừng một tiến trình</i>	3
III.4.	<i>Kích hoạt một tiến trình đang tạm dừng</i>	3
III.5.	<i>Chuyển từ background sang foreground</i>	4
IV.	Một số lệnh có sẵn	4
IV.1.	<i>Kết thúc TinyShell – câu lệnh exit</i>	4
IV.2.	<i>Xem thông tin về các lệnh khác – câu lệnh help</i>	5
IV.3.	<i>Hiển thị thời gian ngày, giờ – câu lệnh date, time</i>	5
IV.4.	<i>Hiển thị thư mục – câu lệnh dir</i>	5
IV.5.	<i>Xem và đặt lại biến môi trường – câu lệnh path/addpath</i>	5
V.	Tín hiệu ngắt từ bàn phím (CTRL + C)	6
VI.	Xử lý theo lô	6
	KẾT LUẬN.....	3
	TÀI LIỆU THAM KHẢO	4

I. Giới thiệu chương trình

I.1. TinyShell

Chương trình TinyShell là một shell dành cho Windows được viết bằng ngôn ngữ C++, theo chuẩn C++14, có một số chức năng cơ bản sau:

- Shell nhận lệnh, phân tích và tạo tiến trình con thực hiện
- Shell chứa các câu lệnh quản lý tiến trình
- Shell hiểu một số lệnh đặc biệt (exit, help, date, time, dir, ...)
- Shell có thể nhận tín hiệu ngắt từ bàn phím để hủy bỏ foreground process đang thực hiện
- Shell có thể thực hiện được file *.bat

I.2. Các thư viện được sử dụng

Trong mã nguồn của chương trình shell sử dụng header windows.h có sẵn của Windows dành cho ngôn ngữ lập trình C và C++ để xây dựng các thao tác xử lý, làm việc với các tiến trình cũng như thao tác với hệ điều hành Windows dựa trên các hàm cụ thể. Ngoài ra, một số thư viện khác cũng được sử dụng nhằm hoàn thành shell như **stdio.h**, **ctime**, ...

I.3. Hướng dẫn cài đặt

Yêu cầu:

- Trình dịch: tốt nhất là gcc $\geq 4.9.2$
- Hệ điều hành: Windows

Chương trình: https://github.com/trinhhungfischer/Project_1_OS_Concept.git

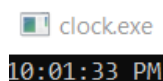
II. Background/Foreground

Chương trình TinyShell có thể thực thi theo hai chế độ: Chế độ hiện (foreground) và chế độ nền (background).

Theo mặc định, mọi tiến trình mà bắt đầu chạy đều là Background Process, tiến trình chạy mà không được kết nối với bàn phím, nếu yêu cầu bất cứ đầu vào từ bàn phím, tiến trình sẽ trong trạng thái Waiting.

Ví dụ:

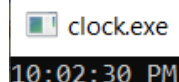
```
myShell> start clock.exe  
myShell>
```



Ở chế độ Foreground, tiến trình nhận lệnh input từ bàn phím và gửi output tới màn hình, Shell phải đợi tiến trình kết thúc, tức là người dùng chỉ có thể giao tiếp với chương trình chứ không thể giao tiếp với TinyShell được nữa. Để chạy chương trình ở chế độ hiện thì thêm foreground ở cuối câu lệnh

Ví dụ:

```
myShell> start clock.exe foreground
```



III. Các câu lệnh quản lý tiến trình

III.1. Liệt kê các tiến trình nền

Dùng lệnh *list* để liệt kê các tiến trình nền đang chạy. Nội dung kết quả gồm nhiều dòng, một dòng tương ứng một tiến trình nền, có dạng:

```
<id> <command> <id process> <status>
```

Trong đó:

- <id> là thứ tự thực hiện của tiến trình
- <command> là câu lệnh shell
- <id process> là chỉ số của tiến trình, dùng để phân biệt với các tiến trình khác

- <status> là trạng thái của tiến trình. Có 2 trạng thái là *running* có nghĩa là tiến trình đang chạy, *stopped* có nghĩa là tiến trình đã bị dừng lại do người dùng gửi tín hiệu *STOP signal* đến tiến trình bằng cách gõ lệnh *stop <id>*

Ví dụ:

```
myShell> list
ID      Cmd Name      Id Process      Status
1       clock.exe     12692           running
2       clock.exe     14732           running
3       countdown.exe 9432            running
```

III.2. Kết thúc một tiến trình

TinyShell cung cấp lệnh *kill* dùng để gửi một tín hiệu (signal) đến một tiến trình (thường dùng để đóng tiến trình). Cú pháp:

```
myShell> kill <id>
```

Ví dụ:

```
myShell> kill 1
Kill clock.exe success
```

III.3. Tạm dừng một tiến trình

TinyShell cung cấp lệnh *Stop* dùng để gửi tín hiệu đến tiến trình và đưa tiến trình về trạng thái *Ready*.

Cú pháp:

```
myShell> stop <id>
```

Ví dụ:

```
myShell> stop 1
Stop clock.exe success
```

III.4. Kích hoạt một tiến trình đang tạm dừng

TinyShell cung cấp lệnh *Resume* dùng để gửi tín hiệu đến tiến trình thông qua Id và đưa tiến trình về trạng thái *Running*.

Cú pháp:

```
myShell> resume <id>
```

Ví dụ:

```
myShell> resume 1
Resume clock.exe success
```

III.5. Chuyển từ background sang foreground

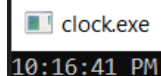
Các tiến trình thường chuyển qua chuyển lại giữa hai trạng thái này trong quá trình hoạt động, việc chuyển trạng thái có thể thực hiện do người dùng, do lệnh từ shell hoặc do lập trình.

Trong bản báo cáo này, chương trình TinyShell cung cấp câu lệnh chuyển đổi tiến trình từ chế độ nền sang chế độ hiện bằng cú pháp:

```
myShell> fg <id>
```

Ví dụ:

```
myShell> fg 1
```



IV. Một số lệnh có sẵn

TinyShell cung cấp một số lệnh để giao tiếp với người dùng dễ dàng hơn.

Project 1	
exit	Quit the myShell.exe program
help	Provide Help information for myShell commands
start	Start new window to run process with background(default) or foreground
date	Display the current date
time	Display the current time
dir	Display a list of files and subdirectories in a directory
list	List of all background processes
stop	Stop a single process with its ID (Know its ID by list command)
resume	Resume a suspended process with its ID (Know its ID by list command)
path	Display all global environment path
addpath	Add a variable to the global environment path
kill	Kill a single process with its ID (Know its ID by list command)

IV.1. Kết thúc TinyShell – câu lệnh *exit*

Để kết thúc Shell, dùng lệnh exit. Cú pháp lệnh như sau:

```
myShell> exit_
```


IV.2. Xem thông tin về các lệnh khác – câu lệnh *help*

Để xem danh sách tất cả các lệnh tích hợp sẵn của TinyShell, dùng lệnh *help*. Để xem thông tin về một lệnh nào đó (ví dụ *time*), dùng lệnh:

```
myShell> help
```

IV.3. Hiển thị thời gian ngày, giờ – câu lệnh *date*, *time*

Để xem thời gian hiện tại của hệ thống, dùng lệnh *date* để hiển thị ngày tháng năm, dùng lệnh *time* để hiển thị giờ phút giây. Ví dụ:

```
myShell> date
Wednesday, April 7, 2021
myShell> time
10:18:27 AM
```

IV.4. Hiển thị thư mục – câu lệnh *dir*

Để hiển thị các file trong thư mục hiện tại, dùng lệnh *dir*. Ví dụ:

```
myShell> dir
Directory
.           <Dir>           07/04/2021    03:25:49
..          <Dir>           07/04/2021    03:25:49
clock.exe   60230 bytes      07/04/2021    03:25:49
countdown.exe 59320 bytes      07/04/2021    03:25:49
myShell.cpp 17483 bytes       07/04/2021    03:25:49
myShell.exe 115737 bytes      07/04/2021    03:25:49
run.bat 67 bytes      07/04/2021    03:25:49
          5 files:           252837 bytes
          2 Dirs
```

IV.5. Xem và đặt lại biến môi trường – câu lệnh *path/addpath*

PATH là một biến môi trường (environment variable) chứa danh sách các thư mục mà chương trình Shell sẽ tìm kiếm cho file thực thi tương ứng với tên lệnh được đưa ra bởi người dùng.

Để hiển thị những đường dẫn nào lưu trong biến môi trường của máy, ta có thể dùng câu lệnh *path*.

Ví dụ:

```
myShell> path
Path=C:\Program Files\Common Files\Oracle\Java\javapath;C:\Windows\system32;C:\Windows\bin;C:\Program Files\Common Files\Intel\WirelessCommon\;C:\Program Files (x86)\NVIDIA Corporation\bin\;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\WINDOWS\System32\OpenSSH\;C:\Program Files\Git\cmd;C:\Users\Dell\AppData\Local\Microsoft\WindowsApps;C:\Users\Dell\AppData\Local\Programs\Microsoft VS Code\bin
```

Để lưu thêm đường dẫn trong biến môi trường, ta có thể dùng câu lệnh *addpath*.

Cú pháp:

```
myShell> addpath C:\User\Dell
```

V. Tín hiệu ngắt từ bàn phím (CTRL + C)

Để hủy bỏ tiến trình đang trong chế độ Foreground, Shell có thể nhận tín hiệu ngắt từ bàn phím bằng cách gõ phím *CTRL + C*.

VI. Xử lý theo lô

Thay vì gõ từng dòng lệnh, người dùng có thể thực thi chương trình thực hiện lần lượt nhiều lệnh khác nhau, những lệnh này được lưu vào trong một file *.bat. Khi một tác vụ chấm dứt thì hệ thống sẽ tự động thực hiện tác vụ tiếp theo mà không cần sự can thiệp từ bên ngoài, do đó hệ thống đạt tốc độ thực hiện cao.

Ví dụ: file run.bat

```
1 start clock.exe
2 date
3 time
4 start countdown.exe foreground
5 kill 1
```

```
myShell> start run.bat
Wednesday, April 7, 2021
09:52:39 PM
Kill clock.exe success
```

KẾT LUẬN

Thông qua việc tự thiết kế và cài đặt một chương trình Shell đơn giản, nhóm em đã hiểu rõ hơn về cách cài đặt và cách thức shell làm việc, nghiên cứu thêm về các API quản lý tiến trình trong Windows.

TÀI LIỆU THAM KHẢO

1. Phạm Đăng Hải. Chương 2 – Quản lý tiến trình. No publisher, 2020