CS360 Ghi chú Bài giảng - Fields

- James S. Plank
- Thư mục: **~ ván / cs360 / ghi chú / Fields**
- Bài giảng: **http://www.cs.utk.edu/~plank/plank/classes/cs360/360/notes/Fields**
- Ghi chú Original: Wed 25 Tháng Tám 1999 11:01:12 EDT
- Sửa đổi lần cuối: Tue 22 tháng 1 năm 2013 14:09:27 EST

Các thư viện trường là một bộ các thói quen mà làm cho đọc đầu vào dễ dàng hơn so với sử dụng **getchar ()** , **scanf ()** hoặc **được ()** . Đây là một thư viện mà tôi đã viết - nó không phải là tiêu chuẩn trong Unix, nhưng nó cũng làm việc với bất kỳ trình biên dịch C (điều này bao gồm trên DOS / Windows). Nếu bạn muốn tham gia lĩnh vực thư viện với bạn sau giờ học, đi trước và làm như vậy. Bạn có thể lấy mã nguồn tại **http://www.cs.utk.edu/~plank/plank/fields/fields.html** .

Để sử dụng các thủ tục trường trong lớp này, bạn nên bao gồm các tập tin **fields.h** , mà có thể được tìm thấy trong thư mục **/ home / ván / cs360 / include** . Thay vì bao gồm tên đường dẫn đầy đủ trong tập tin C của bạn, chỉ cần làm:

```
#include "fields.h"
```

và sau đó biên dịch chương trình với:

```
gcc -I / home / ván / cs360 / bao gồm
```

Khi bạn liên kết các tập tin đối tượng của bạn để tạo một thực thi, bạn cần làm theo các hướng dẫn trong Libfdr ghi chú.

Các **makefile** trong thư mục này làm cả những điều này cho bạn. Khi bạn xem qua các tập tin **printwords.c** , chắc chắn rằng bạn hiểu làm thế nào để biên dịch nó để nó tìm thấy **fields.h** , và do đó các liên kết biên soạn với **libfdr.a** .

Các lĩnh vực định nghĩa thư viện và thực hiện một cấu trúc dữ liệu mà đơn giản hoá chế biến đầu vào trong C. Cấu trúc dữ liệu bao gồm một định nghĩa kiểu và bốn cuộc gọi thủ tục. Tất cả đều được quy định tại **fields.h** :

```
#include <stdio.h>
#define MAXLEN 1001
#define MAXFIELDS 1000

typedef struct {inputstruct
  char * name; / * Tên tập tin * /
  FILE * f; / * File descriptor * /
  int dòng; / * Số dòng * /
```

```
  char text1 [MAXLEN]; / * Các dòng * /
  char Text2 [MAXLEN]; / * Working - chứa các lĩnh vực * /
  int NF; / * Số lượng các lĩnh vực * /
  char * Các trường [MAXFIELDS]; / * Con trỏ trỏ đến các lĩnh vực * /
  int file; / * 1 cho file, 0 cho popen * /
} * IS;

extern LÀ new_inputstruct (/ * FILENAME - NULL cho stdin * /);
extern LÀ pipe_inputstruct (/ * COMMAND - NULL cho stdin * /);
extern int get_line (/ * IS * /); / * Trả về NF, hoặc -1 về EOF. Không đóng
tập tin * /
extern void jettison_inputstruct (/ * IS * /); / * Giải phóng IS và fcloses
các tập tin * /
```

Để đọc một tập tin với các thư viện trường, bạn gọi **new_inputstruct ()** với tên tập tin thích hợp. **New_inputstruct ()** có tên tập tin như là đối số của nó (**NULL** cho đầu vào tiêu chuẩn), và trả về một **IS** như một kết quả. Lưu ý rằng **IS** là một con trỏ đến một **cấu trúc inputstruct .** Đây là **malloc ()** 'd cho bạn trong **new_inputstruct ()** gọi. Nếu **new_inputstruct ()** không thể mở các tập tin, nó sẽ trả về **NULL** , và bạn có thể gọi **perror ()** để in ra những lý do cho sự thất bại (đọc trang người đàn ông trên **perror ()** nếu bạn muốn tìm hiểu về nó).

Một khi bạn có một **IS** , bạn gọi **get_line ()** vào nó để đọc một dòng. **Get_line ()** thay đổi trạng thái của **IS** để phản ánh việc đọc dòng. Cụ thể:

- Nó đặt những nội dung của dòng trong **text1 .**
- Nó phá vỡ dòng thành lời. Các **NF** trường chứa các số từ trong lĩnh vực này. Việc đầu tiên **NF** khe của các **trường** điểm mảng cho mỗi **NF** từ (và những từ này là null-chấm dứt).
- Các **dòng** trường chứa số dòng của các dòng.
- **Get_line ()** trả về **NF** lĩnh vực như giá trị trả về của nó.
- Nó trả về -1 khi nó đến điểm cuối của tập tin.

Jettison_inputstruct () đóng tập tin liên quan **IS** và deallocates (giải phóng) các **IS .** Đừng lo lắng về **pipe_inputstruct ()** bây giờ.

Các thủ tục này rất thuận tiện cho việc xử lý các file đầu vào. Ví dụ, chương trình sau (**printwords.c**) in ra mỗi từ của một tập tin đầu vào thêm vào phía trước với số dòng của nó:

```
#include <stdio.h>
#include <stdlib.h>
#include "fields.h"

main (int argc, char ** argv)
{
```

```
  IS;
  int i;

  if (argc! = 2) {
    fprintf (stderr, "sử dụng: printWords filename \ n");
    xuất cảnh (1);
  }

  là = new_inputstruct (argv [1]);
  if (là == NULL) {
    perror (argv [1]);
    xuất cảnh (1);
  }

  while (get_line (là)> = 0) {
    for (i = 0; i <là-> NF; i ++) {
      printf ("% d:% s \ n", là-> dòng, là-> Trường [i]);
    }
  }

  jettison_inputstruct (là);
  exit (0);
}
```

Vì vậy, ví dụ, nếu tập tin **rex-1.txt** chứa ba dòng sau:

```
Tháng Sáu: Hi ... Anh nhớ em!
Rex: Tương tự ở đây! Em là tất cả tôi có thể nghĩ!
Tháng Sáu: Tôi là ai?
```

Sau đó chạy printWords trên kết quả rex-1.txt ở đầu ra sau đây:

```
UNIX> printWords rex-1.txt
1: Tháng Sáu:
1: Hi
1: ...
1: I
1: bỏ lỡ
1: bạn!
2: Rex:
2: Cùng
2: ở đây!
2: Bạn đang
2: tất cả
2: Tôi
2: có thể
2: suy nghĩ
2: về!
3: Tháng Sáu:
3: I
3: là?
UNIX>
```

Một điều quan trọng cần lưu ý về **fields.o** là *chỉ* **new_inputstruct ()** gọi **malloc () . Get_line ()** chỉ đơn giản điền vào các trường của **IS** cấu trúc --- nó *không* thực hiện cấp phát bộ nhớ. Vì vậy, giả sử bạn muốn in ra từ đầu tiên trên dòng thứ hai-to-cuối cùng. Chương trình sau đây (**badword.c**) sẽ không làm việc:

```
#include <stdio.h>
#include <stdlib.h>
#include "fields.h"

main (int argc, char ** argv)
{
  IS;
  int i;
  char * penultimate_word;
  char * last_word;

  if (argc! = 2) {
    fprintf (stderr, "sử dụng: filename BADWORD \ n");
    xuất cảnh (1);
  }

  là = new_inputstruct (argv [1]);
  if (là == NULL) {
    perror (argv [1]);
    xuất cảnh (1);
  }

  penultimate_word = NULL;
  last_word = NULL;

  while (get_line (là)> = 0) {
    penultimate_word = last_word;
    if (là-> NF> 0) {
      last_word = là-> Trường [0];
    } Else {
      last_word = NULL;
    }
  }

  if (penultimate_word = NULL!) printf ("% s \ n", penultimate_word);
  jettison_inputstruct (là);
  exit (0);
}
```

Tại sao? Nhìn vào những gì sẽ xảy ra khi bạn thực hiện nó trên rex-1.txt:

```
UNIX> BADWORD rex-1.txt
Tháng Sáu:
UNIX>
```

Nó in `` Tháng Sáu: " thay vì `` Rex: " vì **get_line ()** không phân bổ bất kỳ bộ nhớ mới. Cả hai **penultimate_word** và **last_word** kết thúc chỉ để điều tương tự.

Hãy thử một ví dụ khác mà có lẽ là một chút bối rối:

```
UNIX> cat-rex 2.txt
Tháng Sáu: Hi ... Anh nhớ em!
  Rex: Tương tự ở đây! Em là tất cả tôi có thể nghĩ!
Tháng Sáu: Tôi là ai?
UNIX> BADWORD rex-2.txt
ne:
UNIX>
```

Hãy nhớ rằng, **get_line ()** không gọi **malloc () . Malloc ()** chỉ được gọi trong **new_inputstruct () .** Các **lĩnh vực** con trỏ trỏ đến bộ nhớ đó là trong **Text2** một phần của cấu trúc. Như vậy, nếu từ đầu tiên là nhân vật đầu tiên trên đường, sau đó **là-> Trường [0]** điểm để **là-> Text2 .** Nếu từ đầu tiên là nhân vật thứ ba trên đường, sau đó **là-> Trường [0]** chỉ để **là-> Text2 + 2 .** Đó là lý do tại sao các "ne:" được in - vì **penultimate_line** điểm để **là-> Text2 + 2 .** Kể từ **là-> Text2** đã được ghi đè bằng **"June:"** trong **get_line ()** cuộc gọi, **penultimate_line** điểm để "ne:".

Hãy chắc chắn rằng bạn hiểu ví dụ này, bởi vì bạn có thể có được cho mình thành một mớ rắc rối khác. Nếu bạn vẫn gặp vấn đề, đặt trong một số báo cáo in và in ra các địa chỉ của con trỏ.

Các phiên bản chính xác của chương trình là trong **goodword.c** : (lưu ý rằng đây là một chương trình rất hiệu quả vì tất cả các **strdup ()** và **free ()** cuộc gọi Bạn có thể làm tốt hơn nếu bạn muốn..

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "fields.h"

main (int argc, char ** argv)
{
  IS;
  int i;
  char * penultimate_word;
  char * last_word;

  if (argc! = 2) {
    fprintf (stderr, "sử dụng: filename BADWORD \ n");
    xuất cảnh (1);
  }

  là = new_inputstruct (argv [1]);
  if (là == NULL) {
    perror (argv [1]);
    xuất cảnh (1);
  }

  penultimate_word = NULL;
```

```
  last_word = NULL;

  while (get_line (là)> = 0) {
    if (penultimate_word = NULL!) miễn phí (penultimate_word);
    penultimate_word = last_word;
    if (là-> NF> 0) {
      last_word = strdup (là-> Trường [0]);
    } Else {
      last_word = NULL;
    }
  }

  if (penultimate_word = NULL!) printf ("% s \ n", penultimate_word);
  jettison_inputstruct (là);
  exit (0);
}
```

Field.o giả định rằng tất cả các dòng nhập vào ít hơn 1000 ký tự.

---

tailanyf

Bây giờ, như một ví dụ khác, **tailanyf.c** mất *n* như là một đối số dòng lệnh, và in ra cuối cùng *n*

Văn bản gốc

The fields library is a suite of routines that make reading input easier than using getchar() , scanf() or gets() .
Đóng góp bản dịch hay hơn

dòng đầu vào tiêu chuẩn. Nó sử dụng các thư viện trường để đọc đầu vào tiêu chuẩn.