Home

PUBLIC

Stack Overflow

Tags

Users

Jobs

Exercise 1-20 of the book “C - The Programming Language 2nd Edition”

Ask Question

1

Currently I'm reading the book "C - The Programming Language" and I have a questions to this exercise:

"Write a program 'detab' that replaces tabs in the input with the proper number of blanks to space to the next tab stop. Assume a fixed set of tab stops, say every n-colmuns. Should n be a variable or a symbolic parameter?"

Let aside the question in the exercise I wrote this program:

```
#include <stdio.h>

#define COLUMNS 5   /* number of columns for a tab */

int main()
{
    char c;
int i;

while ((c = getchar()) != EOF) {
    if (c == '\t') {
        for (i = 0; i < COLUMNS; i++)
            putchar(' ');
    }
    else
        putchar(c);
}

return 0;
}
```

Then I checked Online for solutions and found this:

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

#define MAX_BUFFER   1024
#define SPACE        ' '
#define TAB          '\t'

int CalculateNumberOfSpaces(int Offset, int TabSize)
{
   return TabSize - (Offset % TabSize);
}

/* K&R's getline() function from p29 */
int getline(char s[], int lim)
{
  int c, i;

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

  return i;
}

int main(void)
{
  char  Buffer[MAX_BUFFER];
  int TabSize = 5; /* A good test value */

  int i, j, k, l;

  while(getline(Buffer, MAX_BUFFER) > 0)
  {
    for(i = 0, l = 0; Buffer[i] != '\0'; i++)
    {
      if(Buffer[i] == TAB)
      {
        j = CalculateNumberOfSpaces(l, TabSize);
        for(k = 0; k < j; k++)
        {
          putchar(SPACE);
          l++;
        }
      }
      else
      {
        putchar(Buffer[i]);
        l++;
      }
    }
  }

  return 0;
}
```

My question now is what the difference between my code an the other code is. I thought the exercise would ask that each \t that occurs should be replaced by n spaces.

Now I don't what the other code does.

Maybe I understood the exercise wrong. If so please explain it to me.

Thanks for your help.

Ok so how about this solution:

```
    #include <stdio.h>

#define TAB 8   /* size of a tab */

int main()
{
char c;
int i,column = 0;

while ((c = getchar()) != EOF) {
    if (c == '\t') {
        for (i = column; i < TAB; i++){
            putchar(' ');
        }
        column = -1;
    }
    else{
        putchar(c);
        }
    column++;
}
return 0;
}
```

New Version:

```
#include <stdio.h>

#define TAB 8   /* size of a tab */

int main()
{
char c;
int i,column = 0;

while ((c = getchar()) != EOF) {
    if(c == '\n')
        column = -1;

    if(column >= TAB)
        column = 0;

    if (c == '\t') {
        for (i = column; i < TAB; i++){
            putchar(' ');
        }
        column = -1;
    }
    else{
        putchar(c);
        }
    column++;
}
return 0;
}
```

c replace tabs space

edited May 10 '14 at 18:04

asked May 10 '14 at 16:00

josepchappa
32 1 7

number of spaces to tab stop will vary depending on the location tab. – BLUEPIXY May 10 '14 at 16:05

2 Answers

1

You don't want to replace every tab by the same number of spaces: some will be replaced by less than that.

Imagine we want TAB to 10 spaces

```
1234567890123456789012345678901234567890
one<TAB>  two<TAB>  fortytwo  four
   ^^^^^^^   ^^^^^^^        ^^
   7 space   7 space        2
```

answered May 10 '14 at 17:10

pmg
85.7k 9 100 174

Look at my last code. This seems to work. – josepchappa May 10 '14 at 17:13

Your last code fails with input like `Pneumonoultramicroscopicsilicovolcanoconiosis<TAB>foobar` – pmg May

10 '14 at 17:21

Now I think I got it. "column" was sometimes not right. And it works with Pneumonoultramicroscopicsilicovolcanoconiosis<TAB>foobar – josepchappa May 10 '14 at 18:07

1

You got it wrong. You have to replace each tab by spaces until the next tab stop. That depends exactly on where the cursor is.

So, if there's a tab stop every 5 columns, and you can think of it like:

```
-----|-----|-----|-----|-----|-----|-----|-----|-----|
```

Where `|` is a tab stop and `-` is a character. Hitting a tab takes you to the next tab stop. So, for example, if you write 3 characters from the beginning of the previous tab stop, and then hit tab, you only have to write 2 spaces.

answered May 10 '14 at 16:08
Filipe Gonçalves
17.5k 5 33 61

I didn't manage to do it in the comments, so I put my edited new code in my question. Would be nice if you have a look at it. – josepchappa May 10 '14 at 17:03