Washington State University Vancouver

CS 360 Precheck

Balanced Parentheses (Using a Stack)

Professor:
Ben McCamish

Overall Assignment

Write a program (in C) targeted at the Linux platform which implements a stack that will be used to determine whether a given file with text possesses balanced parentheses, curly brackets, and square brackets.

Example

Suppose we have a file with the following text:

```
"#include <stdio.h>
 int main() {
     int numbers[20];
     printf("Hello, World!");
     return 0;
 }"
```

If we look only at the parentheses/brackets, we have “(){[]()}”Ẇe can see that this file would then be considered to be balanced in its parentheses. A file is considered to be imbalanced if:

- for any opening parenthesis/bracket there is no corresponding closing parenthesis/bracket
- A closing parenthesis/bracket comes before an opening parenthesis/bracket
- The parentheses/brackets are improperly nested, Ex: “({)}”

Program Interface

Your program must accept the filename as a command line argument as follows

```
./balancedP <filename>
```

After which, your program will look at every character in the file up to the end of the file and print out either “BALANCED” or “UNBALANCED” in response to whether the file contains balanced parenthesis/brackets or not. The program should be written entirely in a single file and all output should go to stdout.

The Structure

This program is to be implemented using a stack. One way to do this is to implement a linked list style node structure with push and pop functions. An example of a structure would be

```
struct node {
    char bracket;
    struct node* next;
};
```

What to turn in (on Autolab):

- A single C file called “balanced.c”