C Program Analysis

This document explains a simple C program, detailing its components, purpose, and execution flow.

The C Code

```c
#include <stdio.h>

int main(void)
{
    char c;
    puts("Freshman: ");
    c = getc(stdin);

    // Note: The original input had 'Y' || "'Y'". Assuming "'Y'" was a typo
    // and correcting it to 'Y' for proper C syntax and logic.
    if (c == 'Y' || c == 'Y')
    {
        puts("Welcome\n");
    }
    else
    {
        puts("Sorry, try again.\n");
    }

    return 0;
}
```

Explanation of Components

1. Standard Library Inclusion (`#include <stdio.h>`)

- `stdio.h` : This is the **standard input/output library** in C . It provides essential functions for handling input (e.g., from the keyboard) and output (e.g., to the screen).
- **Functions used from `stdio.h` in this program:**
  - `puts()` : A function that writes a string to the standard output (usually the console) and automatically appends a newline character.
  - `getc()` : A function that reads a single character from a specified input stream (in this case, `stdin` for standard input, which is typically the keyboard).

2. The `main` Function (`int main(void)`)

- `main` : This is the designated **entry point** where any C application begins its execution .
- `int` **(Return Type)**: The `main` function is declared to return an integer (`int`). This returned value is an **error code**; `0` typically indicates successful execution, while any non-zero value suggests an error occurred .
- `void` **(Arguments)**: The `void` keyword within the parentheses signifies that the `main` function is **not taking any arguments** . (More complex functions can accept arguments, a concept that would be explored later).

Program Logic (Inside `main`)

- `char c;` : This line is for **declaring variables**. It declares a variable named `c` of type `char`, meaning it can store a single character.
- `puts("Freshman: ");` : This statement calls the `puts()` function to display the prompt "Freshman: " on the console, asking for user input.
- `c = getc(stdin);` : This line demonstrates **using variables**. It calls `getc()` to read a single character from the standard input (`stdin`) and assigns that character to the variable `c`.
- `if (c == 'Y' || c == 'Y')` : This is a **conditional statement** (`if-else`).
  - It checks if the character stored in `c` is equal to 'Y' (uppercase 'Y').
  - The condition `c == 'Y' || c == 'Y'` means "if `c` is 'Y' OR `c` is 'Y'".
    - *(Original note: The raw input had `if (C == 'Y' || c == "'Y'")`. The `"'Y'` is syntactically incorrect in C. It has been corrected to `c == 'Y'` assuming it was meant to be a simple character comparison.)*
  - If the condition is true, the code inside the `if` block is executed: `puts("Welcome\n");`
  - `else` : If the condition is false (i.e., `c` is not 'Y'), the code inside the `else` block is executed: `puts("Sorry, try again.\n");`
- `return 0;` : This statement exits the `main` function, returning `0` to the operating system, indicating that the program finished successfully.

Basal Ganglia Loops

Loop	Cortical Involvement	Striatal Component	Pallidal Component	Thalamic Component
Body Movement	Primary motor cortex, Premotor cortex, Supplementary motor cortex, Somatosensory cortex	Putamen	Lateral globus pallidus, Internal segment	Ventral lateral nucleus, Ventral anterior nucleus
Prefrontal	Dorsolateral prefrontal cortex	Anterior caudate	Globus pallidus, Internal segment; Substantia nigra pars reticulata	Mediodorsal nucleus, Ventral anterior nucleus
Limbic	Anterior cingulate cortex, Orbitofrontal cortex, Amygdala, Hippocampus, Temporal cortex	Ventral striatum (nucleus accumbens)	Ventral pallidum	Mediodorsal nucleus