A stack is a type of data structure that follows the Last-In-First-Out (LIFO) property. Based on this, we can write a program with functions to create a stack, push data onto it, pop data from it, and print the contents of the stack. This can be achieved simply using an array. We define a struct pointer that points to a stack for creating and manipulating it. To begin, we'll typedef a struct pointer named 'StackP' and a struct named 'Stack' which will include an array and an index.

In this project, we'll utilize the malloc function to allocate a space in memory for storing a stack, and we'll use a struct pointer to reference it. Following this, the 'InitStack' function can be employed to initialize the stack by setting its length to -1.

The 'PushStack' function enables the insertion of data into the stack. To terminate the process, simply input '999'. Before pushing data, it's crucial to verify whether the stack is already full. If the length of the stack exceeds the maximum size (MaxSize), it indicates that the stack is full.

图片包含 图形用户界面

描述已自动生成

Prior to executing the 'PopStack' function, which involves removing data from the stack, it's equally important to confirm whether the stack is empty. If the length of the stack is less than or equal to -1, it signifies that the stack is empty.

文本

描述已自动生成

Lastly, we have a function to print the stack, displaying the elements from the last added to the first added.

文本

描述已自动生成