Objective

To simulate process creation and destruction when implemented with linked lists.

Specification

The program creates/destroys child processes based on choosing from a menu of choices, where each choice calls the appropriate procedure:

1) Enter parameters
2) Create a new child process
3) Destroy all descendants of a process
4) Quit program and free memory

Assignment

Create a process creation hierarchy as a dynamic array of length n which references the process control blocks (PCBs), indexed 0 to $n-1$.

Each PCB is a structure consisting of two fields:

- **parent**: a PCB index corresponding to the process' creator
- **children**: a pointer to a linked list, where each node contains the PCB index of one child process and a link to the next child in the list

The necessary functions are simplified as follows:

`create()` Prompts for the parent process PCB[p]. The function creates a new child process PCB[q] of process PCB[p] by:

- allocating a free PCB[q]
- recording the parent's index p in PCB[q]
- initializing the list of children of PCB[q] as empty (NULL)
- creating a new link containing the child's index q and appending it to the linked list of PCB[p]

`destroy()` Prompts for the parent process PCB[p]. The function recursively destroys all descendant processes of PCB[p] by performing the following tasks:

- For each element q on the linked list of children of PCB[p]:
  - call `destroy(q)` recursively
  - free PCB[q]
  - deallocate the element q from the linked list

What NOT to do

- Do **not** modify the choice values (1,2,3,4) or input characters and then try to convert them to integers.
- Do **not** submit alternate versions of this assignment from external sites or previous semesters.

What to turn in

- The source code as a file uploaded to Canvas by the deadline.
- Make sure your code compiles with the following online C compiler before submitting: `https://www.onlinegdb.com/online_c_compiler`

Sample Output

```
Process creation and destruction
--------------------------------
1) Enter parameters
2) Create a new child process
3) Destroy all descendants of a process
4) Quit program and free memory

Enter selection: 1
Enter maximum number of processes: 5

Process creation and destruction
--------------------------------
1) Enter parameters
2) Create a new child process
3) Destroy all descendants of a process
4) Quit program and free memory

Enter selection: 2
Enter the parent process index: 0
PCB[0] is the parent of: PCB[1]

Process creation and destruction
--------------------------------
1) Enter parameters
2) Create a new child process
```

```
3) Destroy all descendants of a process
4) Quit program and free memory

Enter selection: 2
Enter the parent process index: 0
PCB[0] is the parent of: PCB[1]  PCB[2]

Process creation and destruction
--------------------------------
1) Enter parameters
2) Create a new child process
3) Destroy all descendants of a process
4) Quit program and free memory

Enter selection: 2
Enter the parent process index: 2
PCB[0] is the parent of: PCB[1]  PCB[2]
PCB[2] is the parent of: PCB[3]

Process creation and destruction
--------------------------------
1) Enter parameters
2) Create a new child process
3) Destroy all descendants of a process
4) Quit program and free memory

Enter selection: 2
Enter the parent process index: 0
PCB[0] is the parent of: PCB[1]  PCB[2]  PCB[4]
PCB[2] is the parent of: PCB[3]

Process creation and destruction
--------------------------------
1) Enter parameters
2) Create a new child process
3) Destroy all descendants of a process
4) Quit program and free memory

Enter selection: 3
Enter the index of the process whose descendants are to be destroyed: 0

Process creation and destruction
```

```
1) Enter parameters
2) Create a new child process
3) Destroy all descendants of a process
4) Quit program and free memory

Enter selection: 4
Quitting program...
```