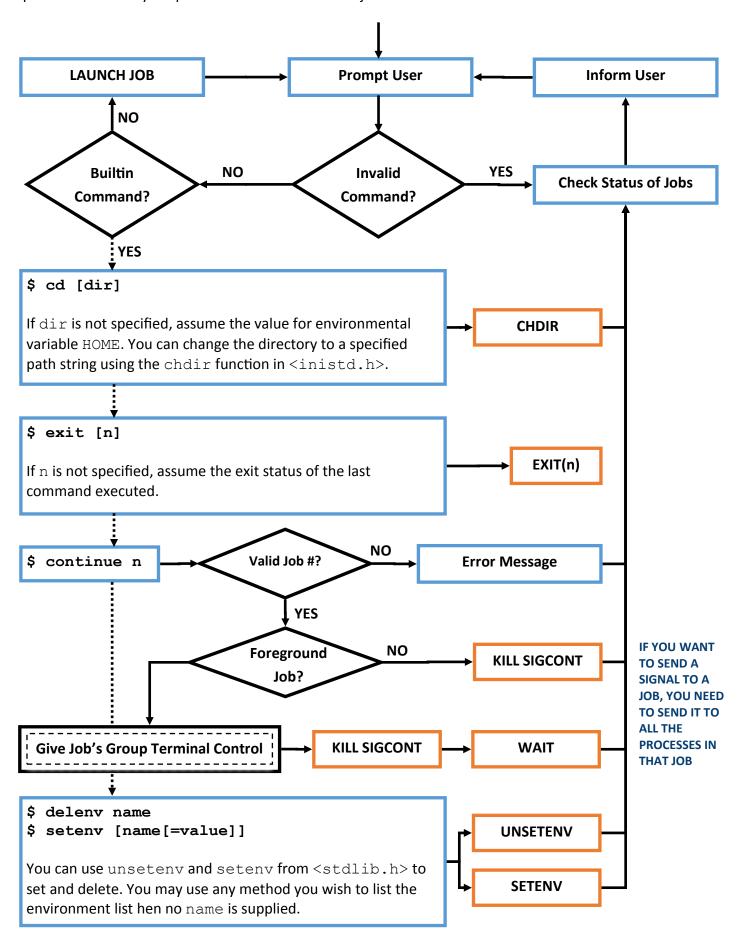
## SIMPLUSH JOB EXECUTION FLOWCHART Simplush This does not include information about the data structures you'll need to implement to actually keep track of the status of each job. **INITIALIZE SHELL** Ignore Job Control Signals ! Become Process Group Leader ! Give Your Group Terminal Control **Execute Builtin Prompt User Inform User** YES **Builtin** NO YES **Invalid Check Status of Jobs** Command? Command? YOU NEED TO **CHECK FOR ANY JOBS THAT HAVE** FOR EACH PROCESS IN THE **TERMINATED OR** PREP FOR LAUNCH **INITIALIZE PROCESS** STOPPED, **UPDATE THEIR Setup Any Pipes** Make First Child a Group Leader STATUSES, AND REPORT THE Place Rest in First's Group **CHANGES TO THE FORK** USER. **Cleanup Any Pipes DOING THIS HERE IS** NO CONVENIENT In Child? **BECAUSE WE** NO **CAN ASSUME** Launched Foreground THE SHELL HAS YES **CONTROL OF THE** Job? TERMINAL. **INITIALIZE PROCESS** YES **NEED TO WAIT FOR ALL PROCESSES IN THE JOB Restore Job Control Signals** WAIT TO REPORT THEIR **CLEANUP STATUS** Make First Proc a Group Leader **Give Your Group** Setup Any I/O Redirection **Terminal Control Update Job Status CONTINUING A STOPPED JOB** NO Part of Foreground **EXEC** EXIT(127) You need to send SIGNCONT Job? **EXIT(127) IS** to the processes in the job. If **ONLY CALLED IF** it's a foreground job, then give **EXEC FAILS** the job's process group con-**FURTHER INITIALIZE PROCESS** trol of the terminal and block Give Job's Group Terminal Control until a status change.

## SIMPLUSH BUILTINS FLOWCHART 1

This does not include information about the data structures you'll need to implement to actually keep track of the status of each job.



## SIMPLUSH BUILTINS FLOWCHART 2

This does not include information about the data structures you'll need to implement to actually keep track of the status of each job.

