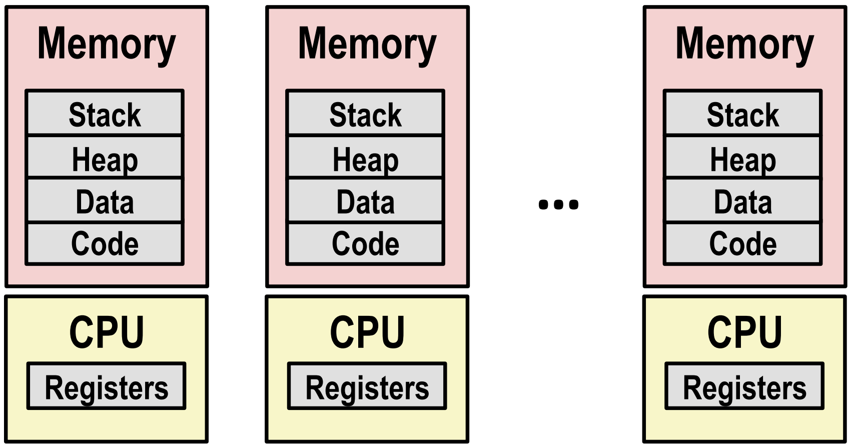
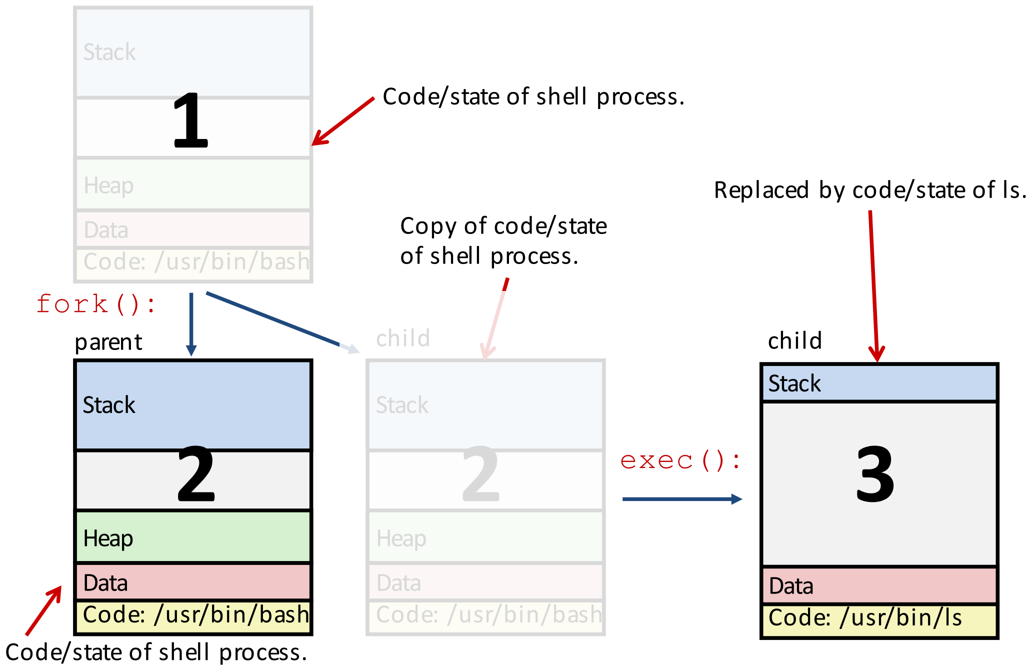
Processes



Fork/Exec

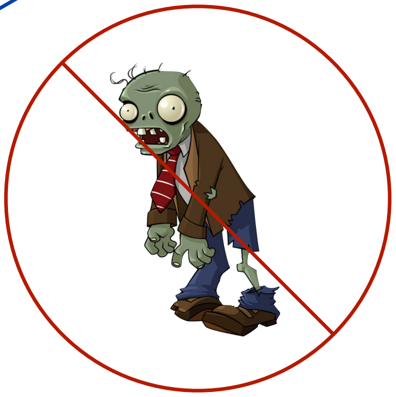


Zombies!

Reaping: waitpid

```
pid_t waitpid(pid_t pid, int* stat, int ops)
```

wait set



Status Macros

```
pid_t waitpid(pid_t pid, int* stat, int ops)
```

`WEXITSTATUS(stat)`	child exit code
`WIFEXITED(stat)`	true if terminated normally (called exit or returned from main)
`WIFSIGNALED(stat)`	true if terminated by signal
`WIFSTOPPED(stat)`	true if stopped (not terminated) by signal

Option Macros

```
pid_t waitpid(pid_t pid, int* stat, int ops)
```

`WNOHANG`	return immediately if child not already terminated
`WUNTRACED`	also wait for stopped children
`WCONTINUED`	also wait for resumed children

System Call Error Handling

Always check return values!

```
    if ((pid = fork()) < 0) {
        fprintf(stderr, "fork error: %s\n", strerror(errno));
        exit(0);
    }
```

Shell Design

```
while (true) {
    Print command prompt.
    Read command line from user.
    Parse command line.
    If command is built-in, do it.
    Else fork process to execute command.
        in child:
            Execute requested command with execv.
                                     (never returns)
        in parent:
            Wait for child to complete.
}
```