```
#include <iostream>
#include <unistd.h>
#include <fcntl.h>
#include <sys/wait.h>
#include <cstdlib>

using namespace std;

int main() {
    int fd[2];
    pid_t pid;

    // Open file11.txt for reading
    int filedes = open("file11.txt", O_RDONLY);
    if (filedes == -1) {
        perror("Error opening file11.txt");
        exit(1);
    }

    // Create a pipe
    if (pipe(fd) == -1) {
        perror("Pipe failed");
        exit(1);
    }

    pid = fork();
    if (pid < 0) {
        perror("Fork failed");
        exit(1);
    }

    if (pid == 0) {
        // Child process

        // Redirect stdin to file11.txt
        dup2(filedes, STDIN_FILENO);
        close(filedes);

        // Redirect stdout to pipe write end
        dup2(fd[1], STDOUT_FILENO);

        // Close unused pipe ends
        close(fd[0]);
        close(fd[1]);

        // Execute sort
        execl("/usr/bin/sort", "sort", (char*)nullptr);

        // If execl fails
        perror("execl failed (sort)");
        exit(1);
    } else {
        // Parent process
        wait(NULL);  // Wait for child to finish

        // Redirect stdin to pipe read end
        dup2(fd[0], STDIN_FILENO);

        // Close unused pipe ends
        close(fd[0]);
        close(fd[1]);
        close(filedes);

        // Execute uniq
        execl("/usr/bin/uniq", "uniq", (char*)nullptr);

        // If execl fails
        perror("execl failed (uniq)");
        exit(1);
    }

    return 0;
}
```

output:**Sample file11.txt:**

nginx

CopyEdit

```
banana
apple
banana
banana
apple
cherry
cherry
banana
```

🧪 Expected Output (after sort | uniq):

nginx
CopyEdit

```
apple
banana
cherry
```

📌 Compile and Run:

bash
CopyEdit

```
g++ main.cpp -o prog
./prog
```