**Example Fork:**

#include<iostream>

#include<unistd.h>

#include<sys/wait.h>

using namespace std;

int main() {

pid\_t pid = fork();

if(pid < 0) {

cerr<<"Fork failed"<<endl;

return 1;

} else if(pid == 0) {

execl("/bin/ls", "ls", "-l", nullptr);

cerr<<"Exec failed"<<endl;

return 1;

} else {

wait(nullptr);

cout<<"Child process completed"<<endl;

}

return 0;

}

**Example 2:**

#include<iostream>

#include<unistd.h>

#include<sys/wait.h>

using namespace std;

int main() {

pid\_t pid = fork();

if(pid < 0) {

cerr<<"Fork failed"<<endl;

return 1;

} else if(pid == 0) {

char \*args[] = {"/bin/ls", "-la", nullptr};

execvp(args[0], args);

cerr<<"Exec failed"<<endl;

return 1;

} else {

wait(nullptr);

cout<<"Child process completed"<<endl;

}

return 0;

}

**Process 1:**

#include<iostream>

#include<mqueue.h>

#include<cstring>

#include<cstdlib>

#include<cerrno>

#include<cstdio>

#define QUEUE\_NAME1 "/test\_queue1"

#define QUEUE\_NAME2 "/test\_queue2"

#define MAX\_SIZE 1024

#define MSG\_STOP "exit"

int main() {

mqd\_t mq1, mq2;

struct mq\_attr attr;

char buffer[MAX\_SIZE];

attr.mq\_flags=0;

attr.mq\_maxmsg=10;

attr.mq\_msgsize= MAX\_SIZE;

attr.mq\_curmsgs=0;

mq1 = mq\_open(QUEUE\_NAME1, O\_CREAT | O\_WRONLY, 0664, &attr);

mq2 = mq\_open(QUEUE\_NAME2, O\_CREAT | O\_RDONLY, 0664, &attr);

if(mq1 == -1 || mq2 == -1) {

std::cerr<<"Error creating queue: "<<strerror(errno)<<std::endl;

exit(1);

}

while(true) {

std::cout<<"Process1, Enter a message: ";

std::cin.getline(buffer, MAX\_SIZE);

if(mq\_send(mq1, buffer, strlen(buffer) + 1, 0) == -1) {

std::cerr<<"Error sending message: "<<strerror(errno)<<std::endl;

exit(1);

}

if(strcmp(buffer, MSG\_STOP) == 0) {

break;

}

ssize\_t bytes\_read = mq\_receive(mq2, buffer, MAX\_SIZE, nullptr);

if(bytes\_read == -1) {

std::cerr<<"error receiving message: "<<strerror(errno)<<std::endl;

exit(1);

}

buffer[bytes\_read] = '\0';

std::cout<<"Process1 received: "<<buffer<<std::endl;

}

mq\_close(mq1);

mq\_close(mq2);

mq\_unlink(QUEUE\_NAME1);

mq\_unlink(QUEUE\_NAME2);

return 0;

}

**Process 2:**

#include<iostream>

#include<mqueue.h>

#include<cstring>

#include<cstdlib>

#include<cerrno>

#include<cstdio>

#define QUEUE\_NAME1 "/test\_queue1"

#define QUEUE\_NAME2 "/test\_queue2"

#define MAX\_SIZE 1024

#define MSG\_STOP "exit"

int main() {

mqd\_t mq1, mq2;

struct mq\_attr attr;

char buffer[MAX\_SIZE];

attr.mq\_flags=0;

attr.mq\_maxmsg=10;

attr.mq\_msgsize= MAX\_SIZE;

attr.mq\_curmsgs=0;

mq1 = mq\_open(QUEUE\_NAME1, O\_CREAT | O\_RDONLY, 0664, &attr);

mq2 = mq\_open(QUEUE\_NAME2, O\_CREAT | O\_WRONLY, 0664, &attr);

if(mq1 == -1 || mq2 == -1) {

std::cerr<<"Error creating queue: "<<strerror(errno)<<std::endl;

exit(1);

}

while(true) {

ssize\_t bytes\_read = mq\_receive(mq1, buffer, MAX\_SIZE, nullptr);

if(bytes\_read == -1) {

std::cerr<<"Error receiving message: "<<strerror(errno)<<std::endl;

exit(1);

}

buffer[bytes\_read] = '\0';

std::cout<<"Process2 received: "<<buffer<<std::endl;

if(strcmp(buffer, MSG\_STOP) == 0)

{

break;

}

std::cout<<"Process2, Enter a message: ";

std::cin.getline(buffer, MAX\_SIZE);

if(mq\_send(mq2, buffer, strlen(buffer) + 1, 0) == -1) {

std::cerr<<"Error sending message: "<<strerror(errno)<<std::endl;

exit(1);

}

}

mq\_close(mq1);

mq\_close(mq2);

mq\_unlink(QUEUE\_NAME1);

mq\_unlink(QUEUE\_NAME2);

return 0;

}

**Receiver:**

#include<iostream>

#include<mqueue.h>

#include<cstring>

#include<cstdlib>

#include<cerrno>

#include<cstdio>

#define QUEUE\_NAME "/test\_queue"

#define MAX\_SIZE 1024

#define MSG\_STOP "exit"

int main() {

mqd\_t mq;

struct mq\_attr attr;

char buffer[MAX\_SIZE + 1];

ssize\_t bytes\_read;

attr.mq\_flags=0;

attr.mq\_maxmsg=10;

attr.mq\_msgsize= MAX\_SIZE;

attr.mq\_curmsgs=0;

mq = mq\_open(QUEUE\_NAME, O\_RDONLY);

if(mq == -1) {

std::cerr<<"Error opening queue: "<<strerror(errno)<<std::endl;

exit(1);

}

bytes\_read = mq\_receive(mq, buffer, MAX\_SIZE, nullptr);

if(bytes\_read == -1) {

std::cerr<<"Error receiving message: "<<strerror(errno)<<std::endl;

exit(1);

}

buffer[bytes\_read] = '\0';

std::cout<<"Received message: "<<buffer<<std::endl;

mq\_close(mq);

mq\_unlink(QUEUE\_NAME);

return 0;

}

**Sender:**

#include<iostream>

#include<mqueue.h>

#include<cstring>

#include<cstdlib>

#include<cerrno>

#include<cstdio>

#define QUEUE\_NAME "/test\_queue"

#define MAX\_SIZE 1024

#define MSG\_STOP "exit"

int main() {

mqd\_t mq;

struct mq\_attr attr;

char buffer[MAX\_SIZE];

attr.mq\_flags=0;

attr.mq\_maxmsg=10;

attr.mq\_msgsize= MAX\_SIZE;

attr.mq\_curmsgs=0;

mq = mq\_open(QUEUE\_NAME, O\_CREAT | O\_WRONLY, 0664, &attr);

if(mq == -1) {

std::cerr<<"Error creating queue: "<<strerror(errno)<<std::endl;

exit(1);

}

std::cout<<"Enter a message: ";

std::cin.getline(buffer, MAX\_SIZE);

if(mq\_send(mq, buffer, strlen(buffer) + 1, 0) == -1) {

std::cerr<<"Error sending message: "<<strerror(errno)<<std::endl;

exit(1);

}

std::cout<<"Message sent: "<<buffer<<std::endl;

mq\_close(mq);

return 0;

}