Q-2 Page - 1

Parent Program.

```
int main(){

    Std::vector<pid_t> kids;
    int AToB[2];        //pipe 1
    int BToA[2];        //pipe 2

    if(pipe(AToB)==-1){
        cerr("Error in creating pipe AToB");
        return 1;
    }
    if(pipe(BToA)==-1){
        cerr("Error in creating pipe BToA");
        return 1;
    }

    child1 = fork();          // forking
    if(child1==-1){
        cerr("Error in creating Process for ProgA");
        return 1;
    }

    if(child1==0){    // Program A

        dup2(AToB[1], STDOUT_FILENO);
        dup2(BToA[0], STDIN_FILENO);
        close(AToB[0]);
        close(AToB[1]);
        close(BToA[0]);
        close(BToA[1]);
        exec(A)          // Start Program A
        cerr("Error in starting program A");
        return 1;
    }
```

Page-2

```
kids.push(child1);

child2 = fork();          // forking
if (child2 == -1){
    cerr("Error in creating Process for Program B");
    return 1;
}

if (child2 == 0){     // program B

    dup2(AToB[0], STDIN_FILENO);
    dup2(BToA[1], STDOUT_FILENO);
    close(AToB[0]);
    close(AToB[1]);
    close(BToA[0]);
    close(BToA[1]);
    exec(B);       // start program B
    cerr("Error in starting program B");
    return 1;

}
kids.push(child2);

close(AToB[0]);       // closing all ends of pipes
close(AToB[1]);          in Parent Process
close(BToA[0]);
close(BToA[1]);
int waitStatus;          // waiting
wait(&waitStatus);
for(kid : kids){
    int status;
    kill(kid, SIGTERM);        // sending signal
    waitpid(kid, &status, 0);
}
```

```
return 0;

}  // End of main (parent) process.
```