System Programming Assignment 4

Goal:

The goal of this assignment is to let students be familiar with the concepts of signals and ICP in Chapter 10 and 14 in the textbook.

Problem Description:

Poker 99 is one popular card game. In this project, we adopt following rules.

RULE:

1. A game has **4** players.
2. Initially, every player has **4** cards in hands, and deck value is **0**.
3. Starting from the **first** player, every player **in turns** chooses a card to discard and add number of the card to the deck value then extract a new card..
4. Once a player **cannot** play without making the total value greater than **99**, this player loses and out the game. The last one that remains in the game is the winner.
5. Function of cards:

Number	Function
A	+1
2	+2
3	+3
4	The order of play is reversed.
5	+5
6	+6
7	+7
8	+8
9	+9
10	+10 or -10
J	Pass
Q	+20 or -20
K	Value is 99 no matter previous deck value.

In this project, you are asked to write three program files.

a. Score_Board.c

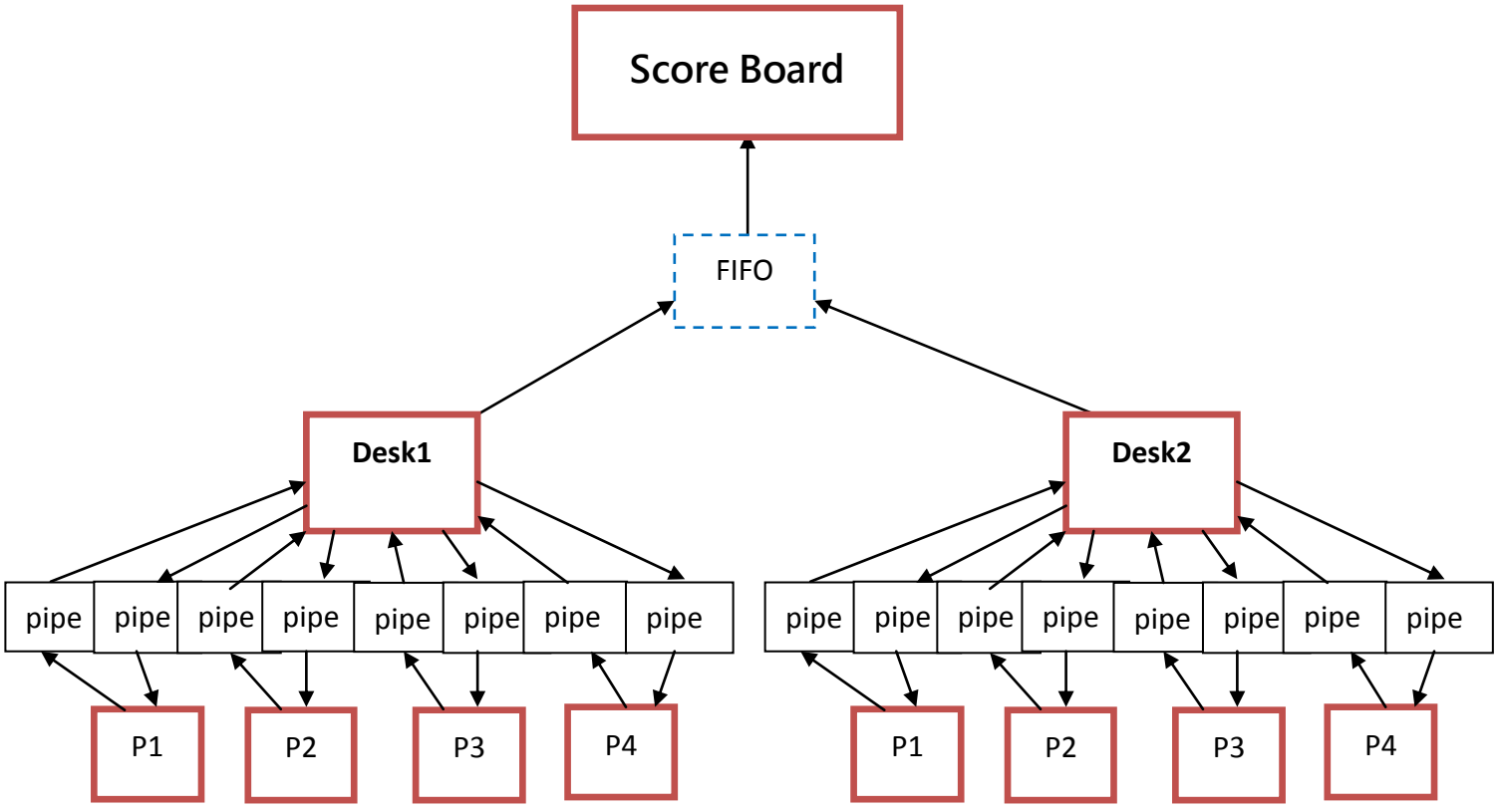
1. Open the well_know_FIFO and then run **background**.
2. Wait for Desk return the winner and write to "./Result.txt".

b. Desk.c

1. Fork 4 players with 4 initial cards.
2. Send current value to one player.
3. Read the card player discard.
4. Examine that weather the player loses or not. If lose, send "FAIL" to player and remove the player from player list. Else, update the deck value, and send a new card to this player then decide the next player.
5. If there only one player in the game, send "WIN\n" to the player and tell Score_Board result. The result send to Score_Board also need to show on stdout.

c. Player.c

1. Examine the card in hand and select one card to discard.
2. If there are still some card can be discarded, player can't suicide.
   Ex: Current deck value is 98, and you have A, 7, 8, 9 in hand.
   You should pick A to keep you in game.
3. After receive card from Desk, return "ACK\n" to Desk.



Flow:

1. Run Score_Board.
2. Run Desk and forks 4 processes and execute Player define in argv.
3. Once 4 players ready, Desk send current value to first player.
4. When a player read a value from Desk, he will select one card and send to Desk. If every card in his hand will make value exceed 99, choosing a card arbitrarily.
5. Desk examines the card player send then check if the player can continue the game. Once player fail, send fail message to player and then close the PIPE. Otherwise send him a new card.
6. When a Player receive a new card, he return a ”ACK”
7. When there is only one player in the game, this Player is the winner and Desk sends “WIN\n” to the player and tell Score_Board result.
8. Score_Board print result to “RESULT.txt” and print on stdout.

Condition:

1. If a player leaves, he loses the game. (ex: use kill to kill player and handle signal)

Protocol:

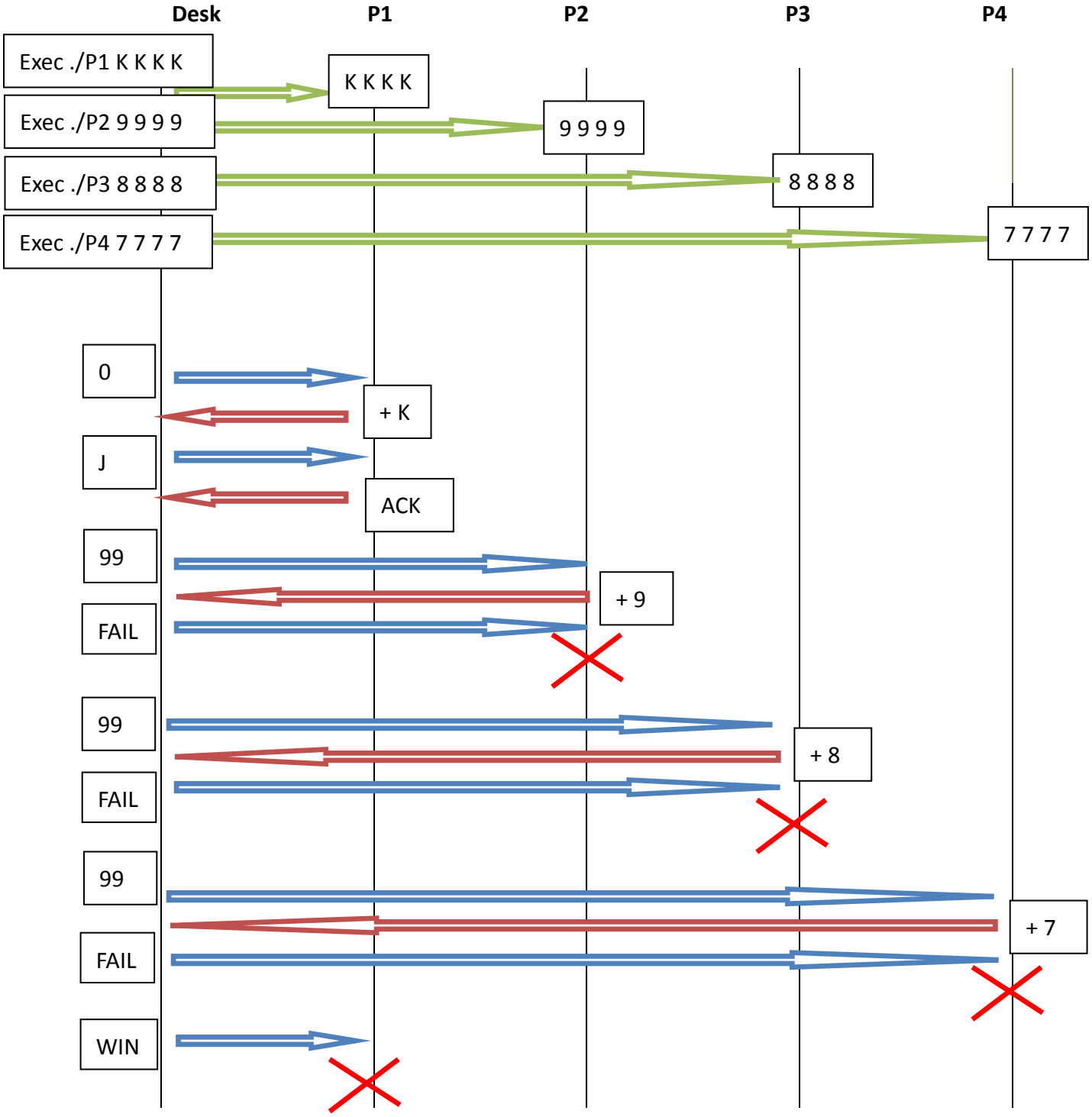
1. Desk to Score_Board: “Desk_ID Player_ID\n”
   Example : “1 1\n”
   notice: player id of first player is 1, the second is 2, …, and so on.
2. Desk tell Player deck value: “value\n”
   Example : “99\n”
3. Desk tell Player new card number: “number\n”
   Example : ”K\n”
4. Desk tells Player lose: “FAIL\n”
   Example : “FAIL\n”
5. Desk tells Player WIN: “WIN\n”
   Example : “WIN\n”
6. Player choose a card and send to Desk: “sign number\n”
   Example : “+ Q\n” **notice: space between sign and number**
7. Player tell Desk he receive new card : “ACK\n”
   Example : ”ACK\n”

Example:

./Score_Board well_know_FIFO &

./Desk 1 P1 P2 P3 P4 well_know_FIFO

Then Desk will fork four process and exec P1, P2, P3 and P4



Then Desk tell Score_Board “1 1”

Implement detail:

1. Suit(花色) of the poker here does not matter the result, thus “Desk.c“ only send players with number.
2. Desk can “randomly” create card for players.

Grading:

Compile Successful: 1pt

Score_Board.c : 1pt

Desk.c: 4pt

Player.c: 2pt

Format:

All source codes(Score_Board.c, Desk.c, Player.c) and Makefile in a directory named by your student id.

use the following instruction to compress:

tar zcvf student_id.tar.gz student_id/

ex:

tar zcvf b99902XXX.tar.gz b99902XXX/

Makefile

```
all:Score_Board Desk Player
Score_Board:Score_Board.o
	gcc -o Score_Board Score_Board.o -lm
Score_Board.o:Score_Board.c
	gcc -c Score_Board.c
Desk:Desk.o
	gcc -o Desk Desk.o -lm
Desk.o:Desk.c
	gcc -c Desk.c
Player:Player.o
	gcc -o Player Player.o -lm
Player.o:Player.c
	gcc -c Player.c
```

Submission:

單班 https://ceiba.ntu.edu.tw/1001sp
雙班 https://ceiba.ntu.edu.tw/1001spcsie

Due:

1/4 (三) 23:59,deduct 5% per day