Q5 (20 points)

As part of a Fall Festival, 10children are participating in a raffle. Each child contributes to the raffle with $10.00/ticket and the total amount is updated. The child receives a ticket[i] that contains a unique number. Out of these number, 5are considered winning numbers. After all children got their tickets, the clown randomly picks a winning number; He searches for who has the same ticket number. The clown should award the winner with 20 dollars. The totalAmount is update. Next, until he runs out of money, he picks another winning number. At the end of the festival, the winners are called. Each winner should leave with 20 dollars.

Shared variable :

winner=0; totalAmount=0; ticket[]; number[]; called[]=F i=1…10 N=10

Child { Clown {

totalAmount += 10; **//starts after all children have their ticket**

get ticket[i]; while(totalAmount != 0) {

while(!call[i]) {B.W.} randomly picks a winner ticket;

winner++; for(int j=1; j<=N; j++) {

if(number on winning ticket==number on

while(winner<5) {B.W.} ticket[j] {

called[j]=True;

Get money(); give the prize; //sleep (random)

Go home(); totalAmount -= 20;

} } // if

} //for

} //while

All child() process and clown() execute concurrently. The following questions are independent one from the other. **PAY ATTENTION TO HOW SHARED VARIABLE ARE ACCESSED.**

• Considering that all children get their tickets and the clown picked each of the five winning numbers. Is it possible for a child who has a **winning ticket** to starve? (Never get out of while(**!called[i]**)); if not explain, if yes, give the execution sequence.

No, it is not possible for a child who has a winning ticket to starve. It is because each child has his ticket[i] that contains a unique number; when the clown randomly picks up a winning ticket, and if the number on winning ticket is equal to the number on the child’s ticket, the child will be called and get the money.

C1: ticket[1]

C2: ticket[2]

C3: ….

Clown:

If(number on winning ticket == number on ticket[1]) {

called[1] = true;

give the prize;

totalAmount -= 20;

}

Child:

C1: While(!Called[1]) {exits}; // because called[1] is true

get money();

go hone();

• Is it possible for the clown to never go home, while condition will never get false? Explain or give the sequence

Yes, because the child has the access to the shared variable totalAmount.

• Under the hypothesis that all winner have been call (**called[j]=Ture**), is it possible for the winning to never leave (**winner<5** never false)? Explain or give the sequence

No, we assume that all winners have been called (called[j] = True), and this means that each child has the ticket containing a unique number. When the clown pick one winning ticket, each child should be called. Hence, it is not possible for the winning number never leave (winner < 5 never false).

• Under the hypothesis that (**winner<5**) will become false, is it possible to have a winner that will not be able to win 20 dollars, because no enough money left? Explain

Yes, it is possible because this is a high level language. The computer must first load the totalAmount, and the totalAmount will be increased by C1 and then stored. However, another thread can load totalAmount before the first thread stores it. There is possibility for data coherence.

C1: totalAmount // load 0

C1: totalAmount+= 10 // inc 10

C2: totalAmount // load 0

C2: totalAmount+= 10 // inc 10

C1: totalAmount = 10 // store

C2: totalAmount = 10 // store