**STORY QUESTION – (難度增加 題目完全不同)**

**Q: Children wait to get their gift. Each child process gets its turn[i] by computing number[i]. A clown has in his hat --- each ball has a different number, from 1 to 10. After all children have their turn set, the clown is --- are the gifts (there are 10 gifts). The clown picks a random ball, and gives the gift to the child whose number is the same with the number of the ball (after that, he throws away the selected ball).**

**Variables:**

**turn[] = 0**

**number[] = 0**

**called[] = 0**

**ballsNumber = 10**

**i = 1,.......10 (N=10)**

**child(int i)**

**{**

**number[i] = 1+max(number[1],number[2],....number[N});**

**turnpi] = number[i];**

**while(!called[i]) {busyWait;}**

**getTheGift(); //sleep**

**ballsNumber--;**

**goHome(); //sleep**

**}**

**clown()**

**{**

**while(ballsNumber>0)**

**{**

**pickABall(); //sleep**

**for(int j=1; j<=N; j++)**

**{**

**if(turn[i]==numberOnBall)**

**{**

**called[j]=true;**

**giveTheGift(); //sleep**

**}//if**

**}//for**

**}//while**

**leave;**

**}//clown**

**\*ALL CHILD PROCESSES EXECUTE CONCURRENTLY.**

**(A) Is it possible for 2 children to compute the same value for number[i]? Explain why and give the execution sequence that can show it.**

Yes it is possible for 2 children to have the same value for number[i]. This is possible because this is a high level language. The computer must first load the number[i], and the number[i] must be calculated then stored. However, another thread can load number[i] before the first thread stores it. There is possibility for data coherence.

P1=loads number[i] = 0

p1=calculates number[i] = 1

p2=loads number[i] = 0

(since p1 did not store number[i], p2 loads what is already in memory)

p2=calculates number[i] = 1

p1=stores number[i] = 1

p2=stores number[i] = 1

**(B) Consider that 6 children already computed their number[i]. Give an execution sequence by which after these 6 children computed their number, the largest computed number[i] is 4.**

P1=loads number[i] = 0

p1=calculates number[i] = 1

p2=loads number[i] = 0

(since p1 did not store number[i], p2 loads what is already in memory)

p2=calculates number[i] = 1

p1=stores number[i] = 1

p2=stores number[i] = 1

p3=loads number[i] = 1

p3=calculates number[i] = 2

p4=loads number[i] = 1

p4=calculates number[i] = 2

p3=stores number[i]=2

p4=stores number [i] =2

p5=loads number[i] =2

p5=calculates number[i] = 3

p5=stores number[i]= 3

p6=loads number[i] =3

p6=calculates number[i] = 4

p6=stores number[i]= 4

**(C) On the hypothesis that each child has a different number[i], is it possible for children to compete for the same give (because turn[i] values are the same)? Explain. If yes, give the execution sequence that will show it.**

No, since each child has a different number, the turns will be a different number as well. Since number is different for each child, and turn[i]=number[i], there is no way for the turns to be the same. It is however possible for the turns to be the same, if the number[i] is the same.

**(D) Consider that at this point, all cihldren have their turn[i] updated (are done with the execution of turn[i]=number[i];) Is it possible for a child to starve (never be called), by busy waiting in the while loop? Explain. If yes, give the execution sequence that will show it.**

No, it is not possible for this to happen. If the children have the same turn[i], they will be called, they will simply compete for the same cookie. If the children all have different number[i], they will all have different number[i] as well, so each one will be called.

**(E) Under the hypothesis that each child had a different turn[i] value and received a gift, is it possible for the clown to not be able to go home because the while condition is still true? Give the sequence that will show this situation.**

Yes, because the children class has access to ballsNumber.

p0=getTheGift();

p0=load ballsNumber = 10

p0=ballsNumber - 1 = 9

p1=getTheGift();

p1=load ballsNumber = 10

p1=store ballsNumber = 9

p0=Store ballsNumber = 9

p0=goHome();

p1=goHome();

This can happen multiple times. If at least two processes access the ballsNumber simultaneously they will load the same value. The same value for ballsNumber is decremented twice. Since the ballsNumber is accessed by the children class and not the clown class, there is possibility for data coherence.