Exercise 4

Student's Name and ID: Tomonori Masubuchi s1240078

Ex.1

Answer the following questions and Explain your answers.

a) What is the meaning of the term Busy Waiting?

Put your answer here.

While a process is in its critical section, any other process that

ties to enter its critical section must loop in the wait statement.

b) What other kinds of waiting are there in an operating system?

Put your answer here.

A process is waiting for an event to happen in some queue and

it does without having the CPU assigned to it.

c) Can busy waiting be avoided altogether?

Put your answer here.

It can be avoided but incurs the overhead associated with putting a process

to sleep and having to wake it up when the appropriate program state is reached.

Ex.2

The Cigarette-Smokers Problems.

Consider a system with three smoker processes and one agent process that play the following roles:

The smoker:

- Each smoker continuously rolls a cigarette and then smokes it.
- To roll and smoke a cigarette, the smoker needs three ingredients: tobacco, paper, and matches.
- One of the smoker processes has unlimited paper, another has tobacco, and the third has matches.
- The smoker smokes a cigarette during indefinite time.

The agent:

- The agent has an infinite supply of all three materials.

- The agent randomly places two of different ingredients on the table.

Scenario:

After placing two ingredients by the agent, the smoker, who has the remaining ingredient, makes the following actions:

1. Picks up two ingredients,
2. Signals the agent.
3. Makes a cigarette and smokes.
4. After smoking finished, the smoker returns to wait ingredients again.

After obtaining a signal from smoker, the agent puts out another two of three ingredients, and the cycle repeats.

Task:

Write a program to synchronize the agent and the smokers processes using the following primitives:

- ***wait (S);*** and ***signal(S); –*** **wait** and **signal** for a semaphore S shown in the textbook and lecture (See, dining-philosophers problem).
- ***RandNum = rand (i,j);*** - Pick a random number **RandNum** from i to j.
- ***Put_tobacco&paper (); Put_tobacco&match (); Put_match&paper*** **–** put two ingredients on table.
- ***Pick_up_match&paper (); Pick_up_ tobacco&paper ();*** ***Pick_up_tobacco&match();*** **–** pick up two ingredients.
- ***Make&Smoke();*** - make and smoke a cigarette.

----------- Agent program -----------------

```
repeat {
Put your agent program here.
int i = 1, j = 3;
int RandNum = rand(i,j);
switch(RandNum){
case 1 : Put_tobacco&paper();
signal(tobacco&paper);
break;
case 2 : Put_tobacco&match();
signal(tobacco&match);
break;
case 3 : Put_match&paper();
signal(match&paper);
break;
}
signal(table);
wait(agent);
} until false // end forever loop
```

```
----------- Smoker 1 program ------------------
repeat {
Put your smoker program here.
wait(tobacco&paper);
wait(table);
Pick_up_tobacco&paper();
signal(table);
signal(agent);
Make&Smoke();
} until false // end forever loop
```

```
----------- Smoker 2 program ------------------
repeat {
Put your smoker program here.
wait(tobacco&match);
wait(table);
Pick_up_tobacco&match();
signal(table);
signal(agent);
Make&Smoke();
} until false // end forever loop
```

----------- Smoker 3 program ------------------

```
repeat {
Put your smoker program here.
wait(match&paper);
wait(table);
Pick_up_match&paper();
signal(table);
signal(agent);
Make&Smoke();
} until false // end forever loop
```

Ex.3
Show, please, that your program for Problem 2 satisfies to the Critical-Section requirements.

○Mutual exclusion→Using semaphores of table satisfies it.

○Bounded waiting→The smoker will Make&Smoke() after it got the item and open table. It satisfies the Bounded waiting

○Progress→Using semaphores of tobacco&paper, tobacco&match and match&paper satisfies the progress.

Put your answer here.