

**East West University**

**Department of Computer Science and Engineering**

**PROJECT REPORT ON**

**“Amherst Candy Factory Problem”**

**Course: Data Structures**

**Course Code: CSE207**

**Semester: Fall 2022**

**Section: 02**

**Under the Guidance Of:**

Nawab Yousuf Ali

Senior Lecturer, Department of Computer Science & Engineering

East West University, Bangladesh.

**Submitted By:**

Sagar Karmoker (ID: 2020-2-60-054)

Md Arnob Tanvir(ID: 2020-1-60-220)

**Date of Submission:** 20-05-2022

**Introduction To Amherst Candy Factory problem**

Amherst candy factory is preparing up for Halloween and has implemented an assembly line to ramp up Halloween candy production. The assembly line hast to be implemented using a bounded buffer producer and consumers. There are two different classes of workers in the factory: producers and consumers. Each consumer thread makes boxes of different candies using the candies generated by producers, while each producer thread develops a specific variety of candy. After making each candy, the producer places it into the bounded buffer. If the buffer is filled, it must wait until one of the slots becomes available. When I candy objects have been retrieved, it fills up a box of various candy.

We must implement a mechanism to coordinate the actions of producers and consumers. The entire number of candy makers, consumers, and varieties is supplied as command line parameters. We must dynamically allocate RAM for data structures based on the input parameter (s).

**Functions used in this project**

**Semaphore:**A semaphore S is an integer variable that can only be accessed via wait() and signal, the two common atomic actions, aside from initialization ().Initial names for the signal() and wait() operations were P(from the Dutch proberen, “to toest”) and V(from verhogen, “to increment”), respectively. Here’s the definition of wait((),

Wait(s){

While(S<=0)

;//busy wait

s- -

}

The definition of signal() is as follows:

Signals(S){

S++;

}

All the modifications of the integer value of semaphore in the wait() and signal() 0perations must be executed indivisibly. So that, No other process may concurrently change the same semaphore value while one process alters it. Furthermore, the examination of the integer value of S(S<=0) and any potential modifications (S- -) must proceed uninterrupted in the case of wait(S).

**Producer consumer problem:** When we try to synchronize many processes, the Producer-Consumer problem is a classic multi-process synchronization problem.

In the producer-consumer conundrum, there is one Producer who creates some goods, and there is one Consumer who consumes those goods. The same fixed-size memory buffer is shared by both producers and consumers. The Producer's job is to create the item, store it in the memory buffer, and then begin creating objects once again. Consuming the item from the memory buffer is the responsibility of the Consumer.

**Producer implementation:**

item nextProduced;

while (true) {

/\* produce an item and put in nextProduced \*/

while (((in + 1) % BUFFER\_SIZE) == out)

; /\* do nothing \*/

buffer[in] = nextProduced;

in = (in + 1) % BUFFER\_SIZE;

}

* Produces an item, wait for buffer space and store in buffer

**Consumer implememtation:**

item nextConsumed;

while (true) {

while (in == out)

; /\* do nothing \*/

nextConsumed = buffer[out];

out = (out + 1) % BUFFER\_SIZE;

/\* consume the item in nextConsumed \*/

}

* Waits for an item available, load it from buffer and consume it.

**Pthreads:** Pthreads is an abbreviation for the POSIX standard (IEEE 1003.1c) that defines an API for thread creation and synchronization. This is a thread behavior definition, not an implementation. It works best on multi-processor or multi-core systems where the process flow may be scheduled to run on a different CPU, improving performance through parallel or distributed processing.

* To use two types of threads. One for producing and one for consuming.

**Reference:** A Silbershatz, P Baer Galvin, G Gagne, “Operating System Concepts”, 9th editions, P: 266, <https://www.javatpoint.com/producer-consumer-problem-in-os>, <https://www.cs.cmu.edu/afs/cs/academic/class/15492-f07/www/pthreads.html>