Tyler Wagner

2/21/23

COSC 3319.01

MWF 9 - 9:50

B Option

**COMMAND LINE:**

C:\Users\tyler\Desktop\ATC\Ada\DSLAB1-2.0\obj>type Data2.txt

30 4 3

C:\Users\tyler\Desktop\ATC\Ada\DSLAB1-2.0\obj>productdistributionmain.exe < Data2.txt > Results

C:\Users\tyler\Desktop\ATC\Ada\DSLAB1-2.0\obj>type Results

**OUTPUT:**

Enter the size of the stack

How many Product Generators?

How many points of sale?

BB delivered.

delivered.

GateKeeper insert accepted TOMATO B

Next grain shipment arrives 1.27761E+00 Time units!

B delivered.

B delivered.

GateKeeper insert accepted RICE B

Next grain shipment arrives 4.34619E-01 Time units!

B delivered.

B delivered.

GateKeeper insert accepted RICE B

Next grain shipment arrives 8.03268E-01 Time units!

GateKeeper insert accepted SQUASH B

Next grain shipment arrives 1.22679E+00 Time units!

M delivered.

GateKeeper insert accepted CORN B

Next grain shipment arrives 3.28606E-01 Time units!

M delivered.

B delivered.

GateKeeper insert accepted SQUASH B

Next grain shipment arrives 2.27514E+00 Time units!

GateKeeper insert accepted FOWEL M

Next grain shipment arrives 8.79319E+00 Time units!

GateKeeper insert accepted FISH M

Next grain shipment arrives 1.02896E+00 Time units!

B delivered.

GateKeeper insert accepted TOMATO B

Next grain shipment arrives 1.03604E+00 Time units!

B delivered.

GateKeeper insert accepted POTATOES B

Next grain shipment arrives 3.06212E+00 Time units!

B delivered.

GateKeeper insert accepted TOMATO B

Next grain shipment arrives 6.55132E+00 Time units!

GateKeeper insert accepted TOMATO B

Next grain shipment arrives 7.17555E+00 Time units!

Meat Sold!

FISH M Removed by GateKeeper for shipment.

Retail Sales successfuly sold FISH M

B delivered.

GateKeeper insert accepted WHEAT B

Next grain shipment arrives 2.07654E+00 Time units!

Meat Sold!

FOWEL M Removed by GateKeeper for shipment.

Retail Sales successfuly sold FOWEL M

M delivered.

GateKeeper insert accepted STEAK M

Next grain shipment arrives 1.30332E+00 Time units!

M delivered.

GateKeeper insert accepted FOWEL M

Next grain shipment arrives 1.90089E+00 Time units!

B delivered.

M delivered.

GateKeeper insert accepted WHEAT B

Next grain shipment arrives 4.94873E+00 Time units!

GateKeeper insert accepted FOWEL M

Next grain shipment arrives 4.29309E+00 Time units!

B delivered.

M delivered.

GateKeeper insert accepted SQUASH B

Next grain shipment arrives 1.03034E+00 Time units!

GateKeeper insert accepted PORK M

Next grain shipment arrives 8.25469E-02 Time units!

M delivered.

GateKeeper insert accepted STEAK M

MNext grain shipment arrives delivered.

2.12869E-01 Time units!

M delivered.

GateKeeper insert accepted FOWEL M

Next grain shipment arrives 1.97321E-01 Time units!

B delivered.

M delivered.

M delivered.

GateKeeper insert accepted FOWEL M

Next grain shipment arrives 3.60628E-01 Time units!

M delivered.

GateKeeper insert accepted POTATOES B

Next grain shipment arrives 3.23951E+00 Time units!

GateKeeper insert accepted FOWEL M

Next grain shipment arrives 1.75448E-01 Time units!

GateKeeper insert accepted STEAK M

Next grain shipment arrives 2.04722E+00 Time units!

M delivered.

GateKeeper insert accepted STEAK M

Next grain shipment arrives 1.90179E+00 Time units!

B delivered.

GateKeeper insert accepted STEAK M

Next grain shipment arrives 1.35851E+00 Time units!

B delivered.

GateKeeper insert accepted POTATOES B

Next grain shipment arrives 1.62639E+00 Time units!

B delivered.

B delivered.

GateKeeper insert accepted WHEAT B

Next grain shipment arrives 3.50942E+00 Time units!

M delivered.

GateKeeper insert accepted CORN B

Next grain shipment arrives 6.13403E-01 Time units!

M delivered.

Rejected by GateKeeper:

RICE B

Rejected = 1. Sent to another distribution facility!

Next grain shipment arrives 2.33375E-02 Time units!

M delivered.

Rejected by GateKeeper:

FOWEL M

Rejected = 2. Sent to another distribution facility!

Next grain shipment arrives 1.31579E+00 Time units!

M delivered.

Rejected by GateKeeper:

FOWEL M

Rejected = 3. Sent to another distribution facility!

Next grain shipment arrives 9.06633E+00 Time units!

B delivered.

Rejected by GateKeeper:

FISH M

Rejected = 4. Sent to another distribution facility!

Next grain shipment arrives 3.71300E+00 Time units!

Rejected by GateKeeper:

STEAK M

Rejected = 5. Sent to another distribution facility!

Next grain shipment arrives 2.88968E+00 Time units!

total meat packets sold 2

total veg packets sold 0

number of veg packets generated 17

number of meat packets generated 13

Hours of operation prior to closing: 40.936156700

B delivered.

B delivered.

M delivered.

**EDITED CODE:**

-- in file GateKeeperService.adb

with Ada.Text\_IO; use Ada.Text\_IO;

with Ada.Calendar; use Ada.Calendar;

package body GateKeeperService is

package IntegerIO is new Ada.Text\_IO.Integer\_IO(Integer); use IntegerIO;

task body GateKeeper is

Capacity : Natural := 0;

function StackSize return Natural is

begin

Put\_Line("Enter the size of the stack");

Get(Capacity);

return(Capacity);

end StackSize;

package twoStacks is new DualStack (Food\_Pack, StackSize); -- \*\* specify size for storage space. \*\* B OPTION HERE

use twoStacks;

meat : Integer :=0;

veg : Integer :=0;

numbersoldmeat : Integer := 0;

numbersoldveg : Integer := 0;

rejected: Integer := 0;

-- Declare food packet counters here.

Start\_Time: Ada.Calendar.Time;

End\_Time: Ada.Calendar.Time;

begin

delay 0.5; -- allow 1/2 hour to initialize facility.

Start\_Time := Ada.Calendar.Clock;

End\_Time := Start\_Time + 1.0 \* 8.0 \* 5.0; -- 1.0 sec./hour \* 8 hours/days \* 5 days

-- Termiate after lossing 5 customers or time to close has arrived.

while rejected < 5 and Ada.Calendar.Clock < End\_Time loop -- Termiate after lossing 5 customers

-- In Ada, a "select" statement with multiple "or" options must uniformly

-- process (randomly) the "accept" statements. This prevents any single

-- "accept" from starving the others from service.

--

-- Rules for "Select":

-- 1) If no task are waiting for service, the task sleeps.

-- 2) If only one of the "accept" entries has a task waiting that task is served.

-- 3) If sleeping and a task or tasks arrive simultaneously, awake a service the

-- the first arrival.

-- 4) If multiple "accepts" have task waiting, service them in random order

-- to prevent starvation.

--

select

-- new arrivals of food

accept acceptMessage( newFood: in Food\_Pack) do

if ( twoStacks.stackAvail ) then

if (getFood\_PackFoodType(newFood) in GrainVegetable) then

twoStacks.pushVeg( newFood);

put("GateKeeper insert accepted ");

PrintFood\_Pack( newFood ); new\_line;

veg := veg + 1;

else

twoStacks.pushMeat(newFood);

Put("GateKeeper insert accepted ");

PrintFood\_Pack(newFood); New\_Line;

meat := meat + 1;

end if;

else

rejected := rejected + 1;

put(" Rejected by GateKeeper: "); new\_line;

PrintFood\_Pack( newFood ); new\_line;

put(" Rejected = "); put(rejected);

put(". Sent to another distribution facility!"); new\_line(3);

end if;

end acceptMessage;

or

-- Accept request for distribution from sales

accept retrieveMessage( newFood: out Food\_Pack; availableForShipment: out Boolean) do

availableForShipment := False;

if not (twoStacks.stackEmpty) then

availableForShipment := True;

twoStacks.popStack( newFood );

PrintFood\_Pack( newFood ); put(" Removed by GateKeeper for shipment."); new\_line;

if(getFood\_PackFoodType( newFood ) in GrainVegetable) then

numbersoldveg := numbersoldveg + 1;

else

numbersoldmeat := numbersoldmeat + 1;

end if;

end if;

end retrieveMessage;

end select;

delay 1.1; -- Complete overhead due to accepting or rejecting a request prior to new iteration.

end loop;

put("total meat packets sold");

Ada.Text\_IO.Put\_Line(Integer'Image(numbersoldmeat)); --gives me total meat sold

put("total veg packets sold");

Ada.Text\_IO.Put\_Line(Integer'Image(numbersoldveg)); --gives me total veg sold

put("number of veg packets generated");

Ada.Text\_IO.Put\_Line(Integer'Image(veg)); --gives total veg generated

new\_Line(1); put("number of meat packets generated");

Ada.Text\_IO.Put\_Line(Integer'Image(meat));

-- print time in service, statistics such as number food pacekets of meat and non-meat products processed.

new\_line(2); put("Hours of operation prior to closing: ");

Ada.Text\_IO.Put\_Line(Duration'Image(Ada.Calendar.Clock - Start\_Time)); new\_line(2);

end GateKeeper;

end GateKeeperService;

-- in file GateKeeperService.ads

--

--

-- The software suite consists of ProductDistributionMain,

-- Food\_DataStructures, Stats\_FoodDistribution, GateKeeperService,

-- Distribution\_Service, CircularQue and Food\_SalesService.

--

-- With the exception of ProductDistributionMain this suite

-- represents the software to manage an "embedded" planetary system

-- food receiving and distribution system. The Distribution\_Service

-- module will be discarded once the embedded software required to

-- manage the physical system is complete and installed.

--

-- The GagteKeeper accepts Food\_Packs frfom inter-galatic transports

-- and repacks them into Food\_Packs suitable for Food\_SalesService to

-- distribute to the planets in the system. Food\_Packs are removed

-- from the default FIFO queue associated with the "accept" statement

-- and stored in a circular queue for distribution.

with Food\_DataStructures; use Food\_DataStructures;

With Stats\_FoodDistribution; use Stats\_FoodDistribution;

with DualStack;

package GateKeeperService is

task GateKeeper is

-- accept Food\_Packs from interplanetary vesssels.

entry acceptMessage( newFood: in Food\_Pack );

--Allow sales to retrive the repackaged Food\_Packs.

entry retrieveMessage( newFood: out Food\_Pack; availableForShipment: out Boolean );

end GateKeeper;

end GateKeeperService;

package body DualStack is

package IntIO is new Ada.Text\_IO.Integer\_IO(Integer);

use IntIO;

--Capacity: Natural := 21;

meatTop: Natural range 0..Capacity := 0;

vegTop: Natural range 0 .. Capacity:= Capacity - 1;

subtype slotindex is natural range 0..(capacity - 1); -- Natural implies >= 0.

box: array(slotindex) of message; -- stack buffer

procedure pushMeat(msg: in message) is

begin

if meatTop < (vegTop - 1) then

meatTop := meatTop + 1;

box(meatTop) := msg;

else

Put\_Line("Error - Stack is full!");

end if;

end pushMeat;

procedure pushVeg(msg: in message) is

begin

if vegTop > (meatTop + 1) then

vegTop := vegTop - 1;

box(vegTop) := msg;

else

Put\_Line("Error - Stack is full!");

end if;

end pushVeg;

function popMeat return message is

begin

if meatTop /= 0 then

meatTop := meatTop - 1;

Put\_Line("Meat Sold!");

return box(meatTop + 1);

else

Put\_Line("Error - Stack is empty!");

end if;

end popMeat;

function popVeg return message is

begin

if vegTop < (21) then

vegTop := vegTop + 1;

Put\_Line("General Product Sold!");

return box(vegTop - 1);

else

Put\_Line("Error - Stack is empty!");

end if;

end popVeg;

procedure popStack(msg: out message) is

begin

if meatTop > 0 then

msg := popMeat;

elsif vegTop < 21 then

msg := popVeg;

else

Put\_Line("Stack empty");

end if;

end popStack;

function stackAvail return Boolean is

begin

if meatTop + (Capacity - vegTop) >= Capacity - 1 then

return false;

else

return True;

end if;

end stackAvail;

function stackEmpty return Boolean is

begin

if meatTop /= 0 or vegTop /= Capacity then

return False;

else

return True;

end if;

end stackEmpty;

end DualStack;

with Ada.Text\_IO; use Ada.Text\_IO;

generic

type message is private;

Capacity: Natural := 0;

package DualStack is

procedure pushMeat(msg: in message);

procedure pushVeg(msg: in message);

procedure popStack(msg: out message);

function popMeat return message;

function popVeg return message;

function stackAvail return Boolean;

function stackEmpty return Boolean;

end DualStack;