

תזכונים ותקשורות

מושגים:

סנכרונייזציה = תקשורת

יערכו תהליך המעביר מידע.

צרוכן= תחילך המקובל מיידע.

בתכניות סדרתיות אין צורך בתזמון מיוחד, אך במספר תהליכי הפעלים בו זמינות יש לעתים צורך להעביר נתונים מטהlixir לטהlixir.

1. תקשורת באמצעות זכרון משותף – בשיטה זו תהליך הנកרא יצרן שולח את המידע שלו לאזרע זכרון משותף, שמננו התהילה שנកרא זכרן ל לוקח את המידע שצירר. הזכרון המשותף הוא **קטע קרייטי** ואין לאפשר לשני תהליכי הימצא בתוכו בו זמנית. שיטה זו יחסית פרימיטיבית, ואני מתאימה למקרה שיש יותר מזכרן אחד ויצרן אחד.

2. **תקשות ישירה – 2** תהליכי מסוגלים ליצור קשר ביניהם. השולח שולח באופן ישיר את המידע למי צריך אותו.

נבחן בין 3 סוגים של תקשורת ישירה:

- סנכרוני
■ חצי סנכרוני
■ סנכרוני מלא

א – סנכרוני: היצahn שלוח אירוע לצרכן, ואינו ממתין לקבלת משוב האם הצרכן מוקן לקבלו או לא. והצריךמושך את האירוע בזמן הפנו.

לדוגמא: סנכרוני – היזרן לא מחייב לזרק שיקבל את האירועabel הרצקן מחייב לארוע,

לדוגמא: הברון ורבורג מילא תפקיד חשוב בהפצת יהדות

בתרגילים שביצעו עד כה, כתבי מהי שיטת הסיכוןיזיה בין התהיליכים:

א. תרגיל הבקבוקים -

ב. תרגיל המחשב והמדפסת –

תור משוכל – קופת חולים

כתב תכנית שמסמכת תור בקופה חולים באופן הבא:

אדם נכנס לקופה החולים בין השעות 8 ל-12 בצהרים, ובוחר (באופן רנדומלי) האם מעוניין לפנות לרופא או לאחות. לכל אחד מהם יש תור משולו, הכנסת אדם לתור לוקחת 10 מיליאניות. טיפול רפואי לוקח 5 שניות וטיפול אחיות לוקח 3 שניות. יש לשים לב לאפשר הכנסה לתור במקרה והוא מלא, אז יש לתת הודעה פלט מתאימה.

המחלקות הדרישות:
1. **שעון** – שעון שספר ומגדם את השעות. בשעה 12 השעון עוצר את עובdotו, המחלקה יורשת מהמחלקה Thread.

המשתנים הדרושים:

זמן התחלת וסיום – המכילים את זמני פתיחת החנות
זמן נוכחי – משתנה שמכיל את השעה בכל רגע (אין הכוונה לשעה אמיתית אלא להשיה של 3000 מיליאניות בין קידום המשתנה בין זמן ההתחלת והסיום).

הmethodות הדרישות:

בונה – מתודה שמאתחלת את זמן ההתחלת והסיום בהתאם למה שנשלח מהתכנית הראשית.

RUN – מתודה שמקדמת את השעה
SOF – מתודה שעוצרת את השעון, מופעלת מה- RUN כאשר מגיעה שעת הסגירה.
2. **תור** – מחלקה שממסמת תור ע"י מערך - יש להעזר במחלקות ELEM, QUEUE, MONITORQUEUE שנלמדו. כדי לשנות את המחלקה ELEM לנוטונים המתאים לאדם.

3. **אחות** – מחלקה שמוציאה מהתור שלה, מבצעת בחירה רנדומלית ממערך מוקן מראש של פעולות לביצוע, והמתנה של 3 שניות.
לדוגמא:

```
String [] peulot={"take blood","zrika",...}
Num=(int)(Math.random()*peulot.length);
```

4. **רופא** – מחלקה שמוציאה מהתור שלה, מבצעת בדיקה – המתנה של 5 שניות.

5. **קופה חולים** – המחלקה הראשית שמבצעת את האובייקטים מהמחלקות השונות:

המתודה MAIN – מבצעת את הפעולות הבאות:

“יצרת ומפעילה את השעון”

“יצרת את התור של הרופא ושל האחות”

“מדמה כניסה אנשים ע”י יצירת האובייקט המתאים, הגרלה לאיזה תור להכנסים, ושולחת למחלקה המתאימה.”

“עוצרת את הפעולות כאשר הגעה השעה 12 והتور ריק.”

תכנות בשפת JAVA

על מנת למשוך את עיקרונות המקבליות נשימוש בשפת JAVA, בה יש מחלוקתה הנקראת Thread אשר מכילה מספר פונקציות בנות אשר משנה את תוכנו בהתאם לכל תכנית. באופן הבא:

- . א. כל מחלוקת שונריצה שתעבד במקביל צריכה לרשות ממחלוקת Thread

לדוגמא: `public class a extends Thread`

- ב. בתוך הבנאי (constructor) של המחלוקת נקרא לפונקציה () start אשר תבצע את התחלין לוישיות התהליכים המסתווים לקבלת זמן מעבד.
- ג. לכל מחלוקת נקבע את הפונקציה () run אשר בה נכתבות את מה שנרצה לבצע כאשר התחלין ייכבל זמן מעבד, בסקרה שלא – הוראות נלט שנות.

ד. הפונקציה () run אשר משמשותה המתנה עד שככל התהליכים העובדים במקביל יסתיימו, עד ניתן לעבר לאיסטרזיה הבאה – נשימוש בשאלות מסוג ה-" Sorcery ". פונקציה זו מוחשבת להלך מרכיב היסל "لتגובה" את המחשב ולক' נכתבה במונח `try-catch`

תורייל הבקטים נפטר בשתי שיטות:

1. שיטה 1 – שליחה של כל המקרים, כולל מקרים קצה אל הפונקציה ובבדיקה האם יש צורך לבצע משוח או שהזהו מקרה קצה בו אין לבצע דבר בתחום הפונקציה – בשפת JAVA המשמעות היא יצירת אובייקט עבור כל מקרה, כולל מקרה קצה, והדבר עיל פחות (אך זה הרבה יותר למתקנת)
2. שיטה 2 – בדיקת מקרים הקצה מתוך התכנית הראשית, ורק במקרה העורק לשולח לפונקציה.

לפניך פתוח עמוד השיטה העשיה, כתבי בעצמך פתרון לפי השיטה הראשונה והגישי בתמיהה של תרגיל הבקבוקים.

GRAPHSET
שיטה לתיאור תוכניתת היכולת ריבוי תהליכיים

סימנים מוסכמים

ערכ

שלב נספר החלב בתוך המוחבר

3

藐חים שונים של שלב:

1

2

3

אחוריאלי

שלב פעיל

שלב לא פעיל

• אטיימון מסמן את השלב הראשי בעת יכול להוות יותר באה. הוא מוצב שלב לשלב בהתאם לתקופות הרציפות.

כל שלב יש צורותנו בזמנו:

א. שלב מסמל את מעב הפעולות: ס=א לא פעיל, נ=א פעיל.
ב. שלב מסמן כמה ומן עבר שחלגת השלב.

פיירות השלב:

לכל שלב יש את הפירות מה יש לבצע בשלב זה

5 הפעלה

ערכ:

מעבר כלפי מעלה עברו למטה)

מעבר כלפי מטה

תמי מטה

תמי מעבר לשלב הבא (התנאי רשות לפיד)

ט

חיק' מעבר:

- כל מעבר שטאו מתקיים ובשלג עלינו היה אטיימון, האטיימון עורך שלב שלאחריו.
- אם יש כמה כלה מלוק מעברים נמקבילים.
- אם שלב מקבל אטיימון והטהאי טליהרוי כבר מתקיים - לסתות זאת השלב מתבצע.

הזהר לא לnikaח תוכנית:

ס=ס א=1 ב=ב

א = משלים על ב

$$a+b = a \text{ Or } b$$

$$a \cdot b = a \text{ And } b$$

$$a+0=a$$

$$a-a'=1$$

$$a \cdot (b+c) = a \cdot b + a \cdot c$$

$$(a+b) ' = a' + b'$$

$$a+1=a$$

$$a+a=a$$

$$a \cdot b = (a+b) \cdot (a+c)$$

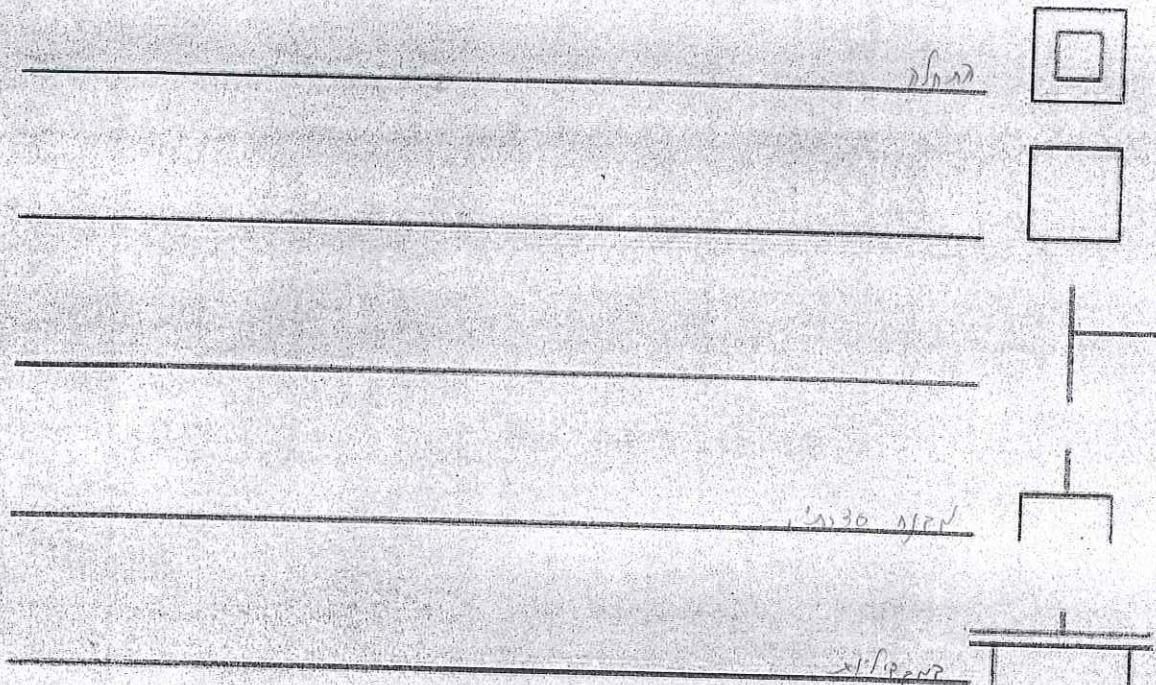
$$(a+b)' = a' + b'$$

$$a+0=0$$

$$a \cdot a=a$$

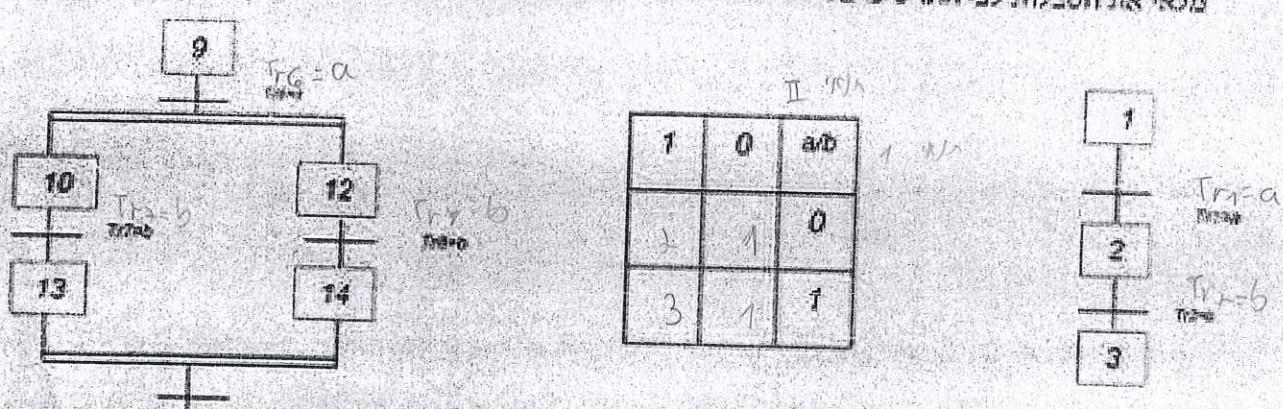
מבנה גראף למערכת דמוי אות - Grafcet

הנדנות:



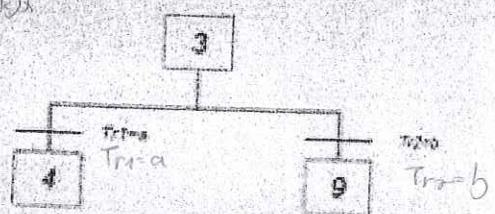
טבלת מעקב ע"פ כללי הלוגיקה הבוליאנית:

מכלאו את הטבלאות לפניהם התרטטיטים.



1	0	ab
12	9	0
14	9	1

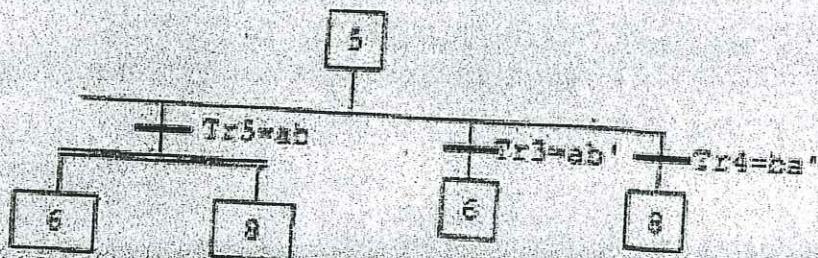
1	0	ab
4	3	0
4	9	1



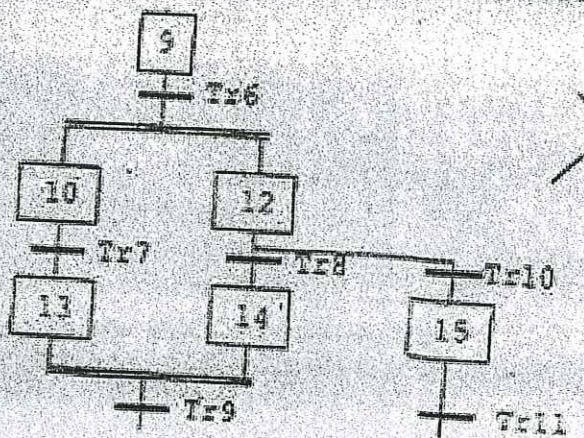
הזרען לזרען נ"ל פ. ברוּם, ר' ליי, סטודנטים

"ערוצו" זרען אמת

אם מתחממים להזען סקוטיל ו- Tr3 מפורש. בניתה שורק אחד כהומואס סטוקים - לבצע אחד סטום, וכשהגאים
מוחקיזיט לבצע את איניהם, ניתן לתאר ביטהה זו:

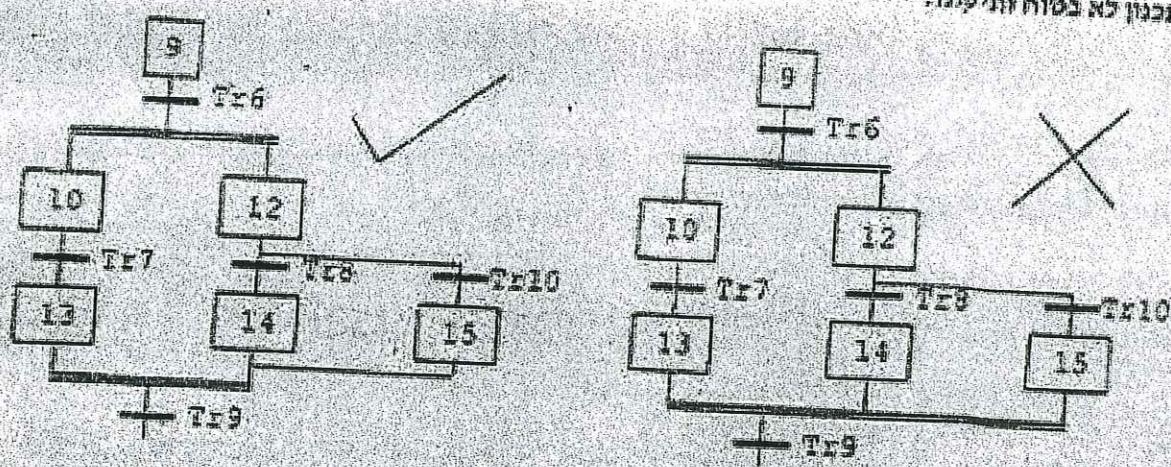


דגם לתכנן לא ביטהה:



ביטהה זו יש יציאת. בתנאים סטוקיים - מונגה תולינים בסקוטיל, ביטהה לא פבוקרת. כדי גוזר מה יש לסייעות
סארכאי מבר-0 Tr10 סטוקים.

דגם לתכנן לא ביטהה וריג'ון:



ביטהה זו יש ריג'ון תולינים שבתובצחים במופטל זכרים יש שליטה.

תור queue

עד כה טיפולנו במקרים בהם קישרנו בין מספר תהליכיים באופן שכל אחד יהיה מעוכב עד לשיום התהליכיים האחרים. נזכיר סוג חדש של טיפול בו כל תהליך הינו עצמאי, והקשר בין התהליכיים מתבצע ע"י תור. תהליך אחד שולח אירועים לתור לפי הצורך, והתהליך השני מוצא מהטור כאשר הוא פניו יוכל לטפל באירוע שנשלח אליו. לדוגמה:

מחשב שולח נתונים להדפסה, המדפסת מדפסה. המחשב יכול לשלוח מספר דפים ברצף, ללא התחשבות בעובדה האם המדפסת פניה או לא כי הדפים נוכנים לתור של הדפסה.

כאשר המדפסת פניה היא שולפת דף מהטור ומדפסה אותו.

נניח כי TOUR הדפים הוא בעל 10 מקומות, למחשב כרגע יש 15 דפים להדפסה, זמן שליחת דף למדפסת הוא 10 מילישניות, זמן הדפסת דף הוא 1000 מ"ש.

א. ציררי גראפט לティאור הבעיה.

בז"ד

- ב. לצורך הרצה במחשב נגדיר פונקציות בסיסיות המימושות את התוור, לאחר הגדרתן מכל ליבא את הקובץ לתוכנות נוספות.

```

import java.io;
class elem
{
int x;
string st;
elem (int xx, string str)
{ x=xx;
st=str;
}
void write ()
{ System.out.printlen (x+" "+st);
}
}

class queue
{
final int max=_____;           אורך התוור;
int tail;
int head;
int count;
elem data []=new elem [max];

public Queue()
{
head=0;
tail=-1;
count=0;
}

int next (int n)
{
return _____;
}

synchronized void EnQueue(elem value)
{
tail next(tail);
data [tail]=new elem(value.x, value.st);
count+=1;
}

```

```
synchronized elem DeQueue()
{
    elem value=data [head];
    head=_____
    count-=1;
    return _____
}
synchronized Boolean IsEmpty()
{ if (count<=0)
    return 1;
else
    return 0;
}
synchronized Boolean IsFull()
{return count>=max;
}

class monitorQueue extends Queue
{
public monitorQueue()
{ super(); _____
}
synchronized void EnQueue(elem item)
{
while (isfull())
try
    {wait(); _____
    } catch (InterruptedException a){}
super.Enqueue(item);
notify();
}
Synchronized elem DeQueue()
{elem item;
while (IsEmpty())
try
{wait(); _____
    } catch (InterruptedException a){}
item=super.deQueue();
notify();
return item;
}
}
```

הקוד של המחשב

```

Class source extends Thread
{
    MonitorQueue fifo;
    Elem item;
    Public source (MonitorQueue f)
    {
        Fifo=f;
        Start();
    }
    Public void run()
    {
        for (int i=1;i<15;i++)
        {try
        { sleep (____);
            catch (InterruptedException a){}
        item=new elem(i, " ");
        fifo.Enqueue(item);
        System.out.print("in");
        Item.write();
        }
        }
    }
}

```

הקוד של המדפסות

```
class destination extends Thread
```

```

{
    MonitorQueue fifo;
    Elem value;
    Boolean stop;
    Public destination (MonitorQueue f)
    {
        Fifo=f;
        Stop=false;
        Start();
    }
    Public void run
    {
        While _____
        {
            _____
            _____
            Value= _____
            _____
            _____
            If (value.x==15)
        }
    }
}
```

```
    Stop=true;  
}  
}  
}  
  
Public class useMonitorQueue  
{  
Public static void main (String args[])  
{  
monitorQueue fifo;  
Source user;  
Destination printer;  
Fifo= _____  
User= _____  
printer= _____  
While (true);  
}  
}
```

"שוני זמן אמת"

בשנים האחרונות, קיימן צורך הולך וגובר במתכנתים מומחמים בתחום מערכות זמן אמת משובצות מחשב, בעלי ידע הן בתוכנה והן בחומרה.
מהן מערכות זמן אמת?

קיימים סוגים רבים של מערכות ממוחשבות. אם נחלקן לפי קритריון של חשיבות זמן התגובה של המערכת, נגיע ל-3 סוגים עיקריים:

1. מערכות אצווה (BATCH SYSTEM) – מערכות סדרתיות, בהן כל פקודה מבצעת לאחר שהקדמת מסתימה.

זמן התגובה: כאלף שניות

בעבר, סוג זה של מחשבים היה נפוץ היוות והמחשבים היו יקרים ורצוליים את העבודה מול המחשב. כמו- המצב שונה, כוח החישוב זול יותר אך לא נשתמש במערכות אצווה אלא אם המערכת תואמת אופי סדרתי.

נשתמש במערכות אלה עבור: כעשרה שניות

2. מערכות מכוננות (ON-LINE)

זמן התגובה: שעון, אך זו אינה תוכנה קרייטית. כיום, רב התכניות הפשטוטות הן מכוננות.

לדוגמה: מכונת כבויים

3. מערכות זמן אמת (REAL TIME SYSTEM) – אלה מערכות הדורשות ביצועים במוגבלות זמן מסוימת, וכך יש חשיבות קרייטית לזמן התגובה, עד כדי כך שתוצאות המכוננות לאחר פרק זמן ארוך ממה שנקבע מראש כדי נחסות כלל נוכנות. למשל רובוט השולח דואר בתוך משרד, אם מערכת העיניים שלו תראה קיר לאחר שהרובוט נתקל בקיר ונחבל אליו המערכת לא מילאה את יעדיה, אפילו שבסופה של דבר זהתה את הקיר.

מערכות הדורשות זמן אמת משובצות במכשירים רבים שאנו מכירים כמו:

טלפון

נבחן בין 2 סוגים של מערכות זמן אמת:

- A. מערכת זמן אמת חזקה – מערכת בעלת דרישות מהיבאות עמידה בזמןני. במערכות כאלה מובטח כי זמן התגובה לא עולה על זמן מסוים, משום שלא ניתן בדרישות עלול להיות תוצאות הרות אסון. לדוגמה:

מערכות **קרייטיות** הן, מערכות תוכנה בהן מחירה של תקלת בזמן עבודה הוא גבוה באופן יוצא דופן. המחיר עשוי להיות אבדן עסקים, אבדן מידע קשה לשחזור ואף פגיעה בחי אדם, בתלות בתחום עסקה של המערכת. במערכות כאלה המהימנות היא לעתים קרובות הדרישה החשובה ביותר מהמערכת.

ישנם שלושה סוגים של מערכות קרייטיות:

- **מערכות בטיחות קרייטית:** מערכות שכשלונן עלול לגרום לפיצעה, אבדן חיים או נזק סביבתי חמור. דוגמא ל מערכת כזו היא מערכות הבקרה במפעלים תעשייתיים גדולים.
- **מערכות משימה קרייטית:** מערכות שכשלונן עלול לגרום לכשלונה של משימה קרייטית. דוגמא ל מערכת כזו היא מערכות הנניות של כלי טיס.
- **מערכות עסקים קרייטית:** מערכות שכשלונן עלול לגרום לפגיעה חמורה בעסקיו של בעל המערכת. דוגמא ל מערכת כזו היא מערכת הגישה לחשבונות של הבנקים.

ב. מערכת זמן אמת חלשה – מערכת בה חשוב שהתהליך יקבל חשיבות של זמן. אך אי עמידה בזמןים היא משמעותית אף לא בעלת תוצאות מסוכנות.

לדוגמה:

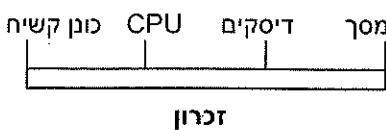
מאפייני מערכת זמן אמת

.1

.2

.3

במחשבים רגילים כל המשאבים משתמשים ומסתמכים על הדיזרין, הקשר הפנימי בין הרכיבים נעשה ע"י אפיקים (BUSES).



במערכות המשובצות במערכות זמן אמת משתמשים בטכניקה אחרת הנקראת SYSTEM CHIP ON - SOC בה המעבד, הדיזרין וכל החומרה נמצאים בשבב אחד קטן. זו טכניקה זולה יותר מאשר הטכניקה הרגילה, והיא טוביה עבור תכניות בעלות נפח קטן.

מערכת זמן אמת תתמוך בדרישות הזמן של שימושת המערכת באמצעות אלגוריתמים שנوتנים לתהליכי החישובים עדיפות גבוהה יותר. האלגוריתמים מנהלי הזמן חייבים לוודא שהתחילה לא יחרוג מהזמן הקצוב לו. אחת הטכניקות לכך היא ע"י הקטנת זמן התגובה לאירועים (כמו פסיקות וכו') למיניהם.

מה מכילה מערכת זמן אמת?

במערכות זמן אמת לא יופיעו רכיבים רבים שאנו מכירים, כמו :

1. מגוון ציוד הקפי – המערכת נועדה עבור מטרה יחידה, כמו לצלם במכשיר, להפעיל מכונת כביסה וכו', לשם כך היא מחייבת לקלט מלחצנים או חיישנים, لكن אין לה צורך בכונן CD או במדפסות.
 2. אפשרות למשתמשים רבים, הרשותות שונות – עבור אבטחת המערכת של כל משתמש, זאת מושם שאפשרויות אלה לוקחות מקום רב הן פיזית והן בזיכרון. לשם השווואה – אם במערכת הפעלה חלונות XP יש 40 מיליון בתים של קוד מקורי, במערכת זמן אמת טיפוסית יש בד"כ כמה אלפי בתים של קוד. לעומת זאת ישנן תוכנות נדרשות במערכת, המשפיעות על תוכנן התוכנות ואופיין: **זמןירות**: יכולתה של המערכת לספק שירות כאשר הוא נדרש, בהסתברות מסוימת. **מרכזית טלפון**, למשל, נדרשת לזמןירות גבוהה מאוד, מכיוון שהALKOH מזכה לשימוש צליל חיוג עם הרמת השופורת.
- אמינות**: יכולתה של המערכת לספק שירות על פי ציפיות המשתמש, בהסתברות מסוימת.

סוגי תקלות הפוגמים באמינות:

- כשל טכני במערכת.
 - טעות אנוש: פועלות משתמש המכניתה אל תוך פעולות המערכת תקלות.
- מערכות תוכנה טובות צופות את טעויות המשתמש ומתקנות אותן לפני ביצוען גורמות לתקלות במערכת.

בטיחות: מידת הבטחון באפשרות של המערכת לגרום לנזק לאנשים או לסביבה.

דוגמאות:

מניעת תאונות: אירוע לא מתוכן המסתויים במוותו / בפיציעתו של אדם / בגין רכוש / לסביבה.

סכנה: מצב בעל פוטנציאלי לגרום להתרחשות תאונה. למשל תקלת בחישון המודד רמות קרינה רדיואקטיבית, יש לתת את הדעת בשלב תכנון הפעולות להסתברות להתרחשות האירועים המבאים לסכנה מסוימת.

כיצד נמנע מיצירת סכנות?

ניתן לתוכנן את המערכת כך שתתקשה מאוד על שימוש לא מכוון או מקרי בה. וידוא כפול, למשל (על ידי לחיצה על שני כפתורי הפעלה במקום אחד, לדוגמה), הופך את האפשרות של הפעלה מקרית של מסור חשמלי למעט בלתי אפשרית.

ניתן לגנות ולהסיר סכנות. תיכנון המערכת צריך לכלול מנגנונים אוטומטיים לגילוי וטיפול בסכנות. בקר המפקח על דוד לחץ יכול להפעיל שסתום חירום לשחרור לחץ ברגע שהלחץ בדוד חוצה סף מסוים, למשל.

להגביל את הנזק. המערכת צריכה להכיל מנגנונים אוטומטיים למזעור הנזק לאחר תאונה. כיום, כל מבנה תעשייתי מודרני מכיל מערכת אוטומטית לכיבוי אש במקרה של שריפה.

אבטחה: מידת הבטחון ביכולתה של המערכת לעמוד בפני חדיות מקריות ומכוונות. דוגמאות: יירוס מחשב, שימוש לא מורשה במשאבי המערכת, שינוי לא מורשה של נתוני המערכת ואף קרייה בלבד...

יש להמנע מיצירת פגימות. ניתן לקחת בחשבון תוכנות ומצבים מועדים לפורענות. מערכת מחשב שאינה נזקמת ואיינה מספקת שירות דרך האינטרנט אינה זקופה לחבר צזה, שלא יוסיף לה דבר מלבד פגימות.

ניתן לגנות ולנטרל התקפות. ניתן לשלב במערכת מערכות מתמחות לגילוי התקפות, כגון אנטי-וירוס, המגלות את ההתקפה ומונטראות אותה לפני שתהופיע לחשיפה.

רצוי להגביל את החשיפה. המערכת צריכה להכיל מנגנונים אוטומטיים למצער החשיפה לאחר התקפה. דוגמא טובה לכך היא מערכות גיבוי אוטומטיות המאפשרות לשחרר את מצב המערכת לפני התקפה.

כולנו מכירים את ההרגשה שטעופת אותנו כאשר המחשב מציג את הודעתה - Program Error, שלאחריה מערכת הפעלה קופת עליינו לסגור את האופיס או לכבות את המחשב, ולשאול את עצמנו ומה שוב פעם לא שמרנו. במחשב האישי, שלנו הנזק יכול להסתכם בשורות האחרונות שכתבנו או באיזה מייל שעוד לא נשלח, אך ישן מערכות שבן כשל פנימי או כתוצאה מהפסקת חשמל "לא בא בחשבון!"

לשם שמירה על יציבות מערכות הפעלה נוצרו מערכות הפעלה בזמן אמיתי Windows Real Time Operating Systems – מערכות הפעלה, הדומות בפועל לנו ל-Windows או לינוקס, שמטרתן העיקרית היא לספק עבודה רציפה ובעיקר צפואה, **شتמנוע נפילות פתאומיות ואי סדרים בפעולות.** הארכיטקטורה הבסיסית של מערכות זמן אמיתי הומצאה כבר ב- 1950 ועדין מהווה את הבסיס למערכות זמן אמיתי המודרניות. כיום המערכות הללו נמצאות בשימוש בכל תחום בתעשייה בו מתבצעת פעילות קריטית.

תרשיים מצב – תרגילים

1. נדמה פעילות של מכשחת דשא בעלת שלושה מצבים:

מצב 0 – מכובה

מצב 1 – גיזום קל

מצב 2 – גיזום כפול

נשים לב כי מצב צמב ניתן לעבור לכל מצב אחר, פעולה המכונה תסתיים במצב מכובה.

2. נדמה פעולה של מכונת כביסה, הפעלתה לפי בחירת תכנית (רגילה או עדינה) ובבחירה רמת טמפרטורה (30, 60 או 90) המכונה מבשת לפי המעלות שנבחרו, סוחטת לפי התכנית שנבחרה, ומסיימת את פעילותה.

3. נדמה פעילות של מכונת מהירות המופעלת ע"י שני פסי דריכה המונחים על הכביש. לב המערכת הוא בקר, תפקידו לפקח על כל שלבי המערכת הכללים קבלת נתוני דריכה, חישוב מהירות נסעה – לפי המרחק והזמן, בדיקת רמת תאורה, הפעלת פלש ומצלמה במידה יש בкарן צורך, כמו כן יש להוציא דו"ח מהירות במקורה ומכוניות צולמה. פסי הדריכה ממתינים מעבר רכב על גביהם אז מתבצעות ארבע דריכות (ארועים) במערכת:

דריכה ראשונה – כshaw גלגלים ראשון עובר על פס דריכה ראשון, מתבצעת דריכה של המערכת

דריכה שנייה – כshaw הגלגלים הראשון עובר על פס הדריכה השני, יתבצע אומדן לפי משקל באיזה רכב מדובר (אנו נטפלרכב פרטיא בלבד)

דריכה שלישית – כshaw הגלגלים השני עובר על פס הדריכה הראשון, מתחילה לעבוד השעון.

דריכה רביעית – כshaw הגלגלים השני עובר על פס הדריכה השני – אז מחושבת מהירות, אם יש צורך מצולמת המכונית ונסלח הדו"ח.

החליטי מתי כדאי לבדוק את רמת התאורה וشرطיו תרשימים מצב מתאים.

תרגיל מסכם

מפעל לייצור פופקורן עובד בצורה הבאה:
בתחלת פעולתו ממלאים 1 טון גרגרי פופקורן. ישן 3 עמדות המקבלות בכל פעם 1 קילו גרגרים, מבלשות, אורזות ומכניות לטור מוכנים. התור הוא בן 30 מקומות, אך מתפרק בקבוצות של 5, כלומר כאשר יש 5 אוריינות מוכנות הן יוצאות מהטור ונארזות יחדיו. המפעל מסיים את פעולתו כאשר נסתיימה אריזת כל הקבוצות.

המחלקות הדרשיות:

1. תור – מחלקה שמשמשת בתור ע"י מערך – יש להעדר במחלקות ELEM, QUEUE, MONITORQUEUE שלמדו. כדי לשנות את המחלקה ELEM לנוטים המתאים לחבילת פופקורן, כמו כן ניתן לשנות את הפונקציה DEQUEUE להזאת 5 חבילות בבת אחת ואת הפונקציה YSEMPYTY לבדיקת הימצאות 5 חבילות בתור.
2. בישול – יירושת מתהילר – עמדה המתקבלת 1 קילו, מבלשת ואורזת – הכל בהודעת פלט. כמו כן מכינסה לתור כל חבילה מוכנה.
המשתנים הדרושים:
 - תור – משתנה מסווג המחלקה MONITORQUEUE.
 - מספר סידורי של העמדת – עברו פלט

הנתודות הדרשיות:

RUN – במתודה זו מבצעים לוליה המתקבלת 1 קילו, מבלשת ואורזת – הכל בהודעת פלט. כמו כן מכינסה לתור כל חבילה מוכנה, כלומר מבצעים ENQUEUE עם המשתנה של 5 שנויות לכל חבילה שנכנסה – הכל בלולאה המסתמימת כאשר המונה של החבילות מגיעה ל-10000.

3. מחלקת אריזה – יירושת מתהילר.

המשתנים הדרושים:

- תור – משתנה מסווג המחלקה MONITORQUEUE.

הנתודות הדרשיות:

RUN – במתודה זו מבצעים לולאת (WHILE) אשר מוציאה מהטור בקבוצות של 5 חבילות, ואורזת כל קבוצה חבילות ייחודי. SOF – עוצרת את פעולה המחלקה, מופעלת מהתכנית הראשית לאחר JOIN של המחלקה בישול.

4. MAIN – מתודה זו מבצעת את הפעולות הבאות:

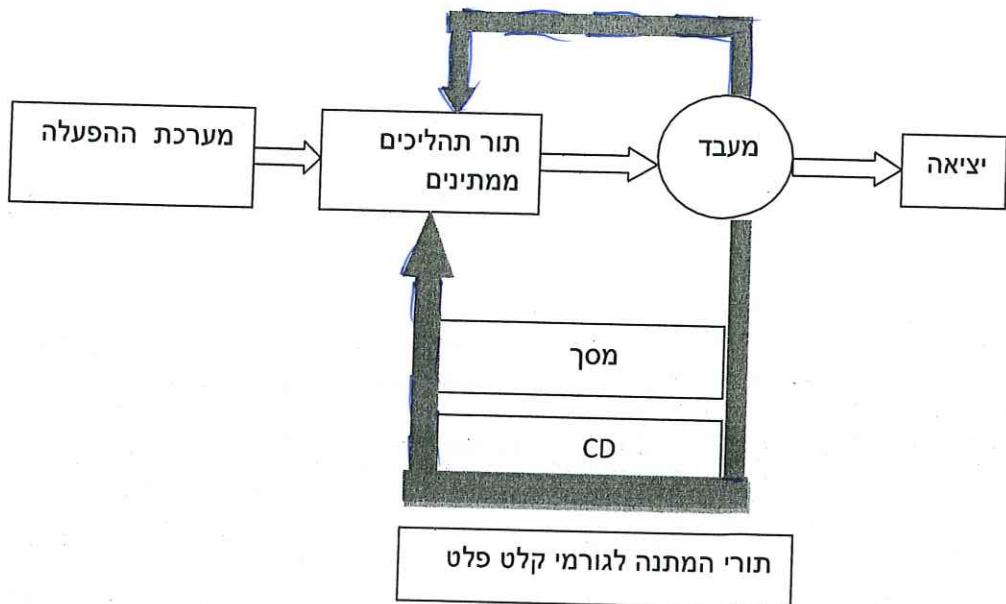
מגדירה את התור

يוצרת ומפעילה את 3 העמדות.

يוצרת ומפעילה את מחלקת האריזה.

יש לבצע () JOIN עבור מחלקת בישול ולאחריו לשלו לפונקציה SOF של מחלקת אריזה לבדיקת סיום פעילותה.

מציג תרשימים בסיסי לפעולת הסדרן, אמנם הסדרן עצמו אינו נראה בתרשימים אך הוא המנהל את פעולהת המערכת "מאחורי הקלעים":



התרשימים מציג מודל בסיסי של מערכת הפעלה, בו רואים מספר טורים בהם נמצא התהיליך בזמן שאינו ב-CPU. סדרן הזמן הוא האחראי על ניהול ה-CPU ועביר את התהיליכים מתור עד לשיטם פעולותם.

כל תהיליך נכנס למערכת הפעלה לתור הממתינים בזמן CPU, כאשר מגיע הזמן נכנס למעבד. התהיליך יצא מהמעבד באחת מהאפשרויות הבאות:

- א. אם נגמר התהיליך (המשימה הושלמה) - יציאה
- ב. אם עבר פלח הזמן אולם לא נגמר התהיליך – יחזיר לתוך תהיליכים הממתינים לריצועzeit.
- ג. אם ציריך נתונים מחומרה – קלט/פלט, מסך מדפסת וכו' – ילך לתוך המתנה לגורם קלט פלט שונים, לאחר מכן יחזיר לתור ממתינים ל-CPU.

מה לדעתך יקרה במצב יש כרגע תהיליך אחד בלבד הזקוק בזמן CPU ארוך יותר מריצועzeit הזמן?

מערכת הפעלה היא שקבועה, על פי הנתונים המוצבים בה, איך עדיפות יש לתהיליך, והאם עליינו לקבל פלח זמן גדול יותר מהאחרים.

❖ רכיב חומרה הכרחי במערכות זמן אמת הוא שעון (TIMER)
 ישנו שעון פנימי, הקובע את קצב העבודה של המחשב
 ישנו שעון חיצוני, השולח פסיקות בקצב קבוע, כך יכול הסדרן לתת לכל תהיליך ריצועzeit זמן זהה בבדיקה לריצועzeit הזמן שהקצתה לתהיליך אחר.

מודל חלוקת הזמן – TIME SHARING

מערכת הפעלה היא תוכנה שנועדה להקל על ההתקשרות בין המשתמש (USER) ובין חלקים שונים במחשב.

ישנם שלושה סוגים עיקריים של מערכות הפעלה:

- א. מערכת לעיבוד יחיד – מערכת המטפלת בתהליכיים בצורה סדרתית, פועלת על מעבד יחיד.
- ב. מערכת ריבוי תכניות – פועלת על מעבד יחיד, בה נסוקן
- ג. מערכת מבוזרת – פועלת במספר תחנות מעבדים שונים בו זמנית, כך פעולות העיבוד שציריך לבצע מתחלקות בין המעבדים.

היות ואי אפשר להפעיל מספר תהליכיים בו זמנית במערכת שיש בה רק מעבד אחד, נשתמש בסדרן הזמן, שתפקידו לחלק את המשאבים הנדרדים בין התהליכיים השונים תוך חסכוּן מירבי בזמן.

הסדרן נותן לכל תהליך להשתמש במעבד לפרקי זמן קצר, כך נוצרת אשליה שתהליכיים מתרחשים במקביל. אורך כל פרק זמן (TIME SLICE) תלוי בחומרה ובסדרן.

במצב בו לכל תהליך יש עדיפות שונה בוחר הסדרן את התהליכיים בעלי העדיפויות הגבוהה יותר שיקדימו לפעול.

לכל תהליך במערכת חייב להיות מקום לפרטיו האישיים כגון עדיפותו, מצבו (פעיל/מושהה), אחסון מידע שעובד לגביו, ועוד

הסדרן פועל על פי מספר אלגוריתמים, אחד מהם מכונה תזמון עם משוב לפיו כל תהליך מקבל רצעת זמן עיבוד קצרה וקבועה, במידה והתהליך לא סיים את פעולתו ברצעת הזמן הוא מוחזר לתור, ומהידע לגביו נשמר בד"כ במחסנית עד לפעם הבאה שיקבל רצעת זמן.

כמה זמן יאלץ תהליך להמתין עד לקבלת זמן עיבוד נוסף?

נניח ונכנה כל רצעת זמן Q, ויש N תהליכיים: כל תהליך ימתין מקסימום – Q*(1-N)

כיום, ישנו שני סוגים עיקריים של אלגוריתמים לסדרנים:

סדרנים סטטיים – בתחילת הביצוע של התכנית נקבעים סדרי העדיפויות של כל התהליכיים, העדיפות הגדולה ביותר, בד"כ ניתנת לתהליך שצורף זמן CPU קצר (כדי שלו יוציאו "פקקים").

סדרנים דינמיים – הסדרן פועל בזמן עבודה המערכת ומתעדכן במהלך ביצוע התכנית. אחד האלגוריתמים הנפוץ ביותר לסדרן דינמי הוא קדימות לתהליכיים שהמועדם האחרון שלהם הם הקרובים ביותר לאחור.

תרגיל מעבדה - לכבוד פורים!

נמש מפעל המכין אוזני המן לפי בחירה של 3 מיליון (פרג, שוקולד או תמרים), לאחר מכן אוזן בחבילות של 0.5 קילו. מוסף 1 מוקפלת ו-2 מטבעות שוקולד ועוטף כשלוח מנות.

בחירת מילוי אוזני המן תעשה בצורה רנדומלית באופן הבא:

Num=(int)(Math.random();

ולאחר מכן קבעי- אם הוגרל 1 אז בצע מילוי פרג וכו'

א. צייר גרפוסט המתאים באופן מדויק את נתוני המפעל.

ב. כתבי אלגוריתם מילולי.

בצעי בשיטת האלגוריתם המילולי השני, בשיטה זו התכנית הראשית בודקת לאילו פונקציות עליה לשלוח לפי מספר האיתרציה. בדיקת מקרי קצה בתכנית הראשית.

ג. שימי בתקיה לבדיקה, הגישי את ציור הגרפוסט.

בהצלחה

תרגיל – דמי מושך

תכנים מושך הופעל בזמן אמיתי. במושך יש מנהל ו-3 מזכירות, כולם פעילים במקביל.

תפקיד המנהל הוא הדפסת הودעה כל פרק זמן קבוע של 6 שניות.

תפקיד המזכירות הוא הגרלת פעולה מתוך מערכת פעולה, כל מזכירה מקבלת לפונקציה הבונה: שם, כמות פעולות לביצוע, השהיה.

בנוסף מוגדר מערכת פעולה לכל המזכירות:

- שליחת פקס
- שיחת טלפון
- קביעת פגישה
- ועד כרצונן

המושך מתחילה את פעילותו בשעה 09:00 ומסיימת את פעילותו בשעה 13:00

השתמשי במחלקה השעון מתרגילים קודמים.

א.شرطיגרפסט לתאזר פעילות המושך

ב. כתבי תכנית בשפת JAVA לטיור המושך

ג. שימי בתיקיה להגשה.