Algorytmy

Bożena Woźna Szcześniak

30 listopada 2016

1 Pakiet `algorithmic`

Stosując pakiet `algorithmic` rozwiąż poniższe zadania.

1.1 Zadanie 1 - napisz kod generujący poniższy algorytm

```
Require: tablica A o rozmiarze n {A=[0,...,n-1]}
Algorytm: MIN
$min = 0$;
for all $i = 1$ to $n$ do
  if $A[i] < A[min]$ then
    $min = i$;
  end if
  $i = i + 1$;
end for
return $min$;
```

1.2 Zadanie 2 - napisz kod generujący poniższy algorytm

```
Stack-Empty(S)
if $top[S] = 0$ then
  return $true$
else
  return $false$
end if
Pop(S)
if Stack-Empty(S) then
  error "niedomiar"
end if
$top[S] = top[S] - 1$
return $S[top[S] + 1]$
```

1.3 Zadanie 3 - napisz kod generujący poniższy algorytm

```
   List-Search(L,k)
1: x := head[L];
2: while (x!=NIL and key[x]!=k) do
3:    x := next[x];
4: end while
5: return x;
```

2 Pakiet `listings`

Stosując pakiet `listings` rozwiąż poniższe zadania. Kod programu napisany jest w Javie.

2.1 Zadanie 1 - napisz kod generujący poniższy kod

```java
import java.util.Random;
public class AltBitProtocol {
  public static void main(String[] args) {
    LossyChannel channel = new LossyChannel();
    (new Thread(new Sender(channel))).start();
    (new Thread(new Receiver(channel))).start();
  }
}
class LossyChannel {
  private boolean protocolBit, channelEmpty = true, ackBit = true;
  private Random r;
  public LossyChannel() {r = new Random();}
  public synchronized boolean getAckBit() {notify(); return ackBit;}
  public synchronized void putAckBit(boolean ackBit) {
    this.ackBit = ackBit;
    int ignoreBit = r.nextInt(2);
    if (ignoreBit > 0) {this.ackBit = !this.ackBit;} notify();
  }
  public synchronized boolean get() {
    while (!channelEmpty) {try {wait();} catch (InterruptedException e){}}
    channelEmpty = true; notify();  return protocolBit;
  }
  public synchronized void put(boolean protocolBit) {
    while (!channelEmpty) {try {wait();} catch (InterruptedException e){}}
    int ignoreBit = r.nextInt(2);
    channelEmpty = false; if (ignoreBit > 0) channelEmpty = true;
    notify();
  }
}
class Sender implements Runnable {
  private LossyChannel channel;
  public Sender(LossyChannel channel) {this.channel = channel;}
  public void run() {
    boolean protocolBit = false; Random r = new Random();
    while (true) {
      if (protocolBit != channel.getAckBit()) channel.put(protocolBit);
      else protocolBit = !protocolBit;
      try {Thread.sleep(r.nextInt(1500));} catch (InterruptedException e){}
    }
  }
}
class Receiver implements Runnable {
  private LossyChannel channel;
  public Receiver(LossyChannel channel) {this.channel = channel;}
  public void run() {
    Random r = new Random();
    while (true) {
      boolean protocolBit = channel.get(); channel.putAckBit(protocolBit);
      try {Thread.sleep(r.nextInt(1500));} catch (InterruptedException e){}
    }
  }
}
```

2.2 Zadanie 2 - napisz kod generujący poniższy kod

Z powyższego kodu wyświetl tylko linie od 1 do 8.

```
import java.util.Random;
public class AltBitProtocol {
  public static void main(String[] args) {
    LossyChannel channel = new LossyChannel();
    (new Thread(new Sender(channel))).start();
    (new Thread(new Receiver(channel))).start();
  }
}
```

2.3 Zadanie 3 - napisz kod generujący poniższy kod

Dane są kolory jak na wykładzie. Zdefiniuj własny styl. aby wyglądał jak ten poniżej.

```
import java.util.Random;
public class AltBitProtocol {
  public static void main(String[] args) {
    LossyChannel channel = new LossyChannel();
    (new Thread(new Sender(channel))).start();
    (new Thread(new Receiver(channel))).start();
  }
}
class LossyChannel {
  private boolean protocolBit, channelEmpty = true, ackBit =
    true;
  private Random r;
  public LossyChannel() {r = new Random();}
  public synchronized boolean getAckBit() {notify(); return
    ackBit;}
  public synchronized void putAckBit(boolean ackBit) {
    this.ackBit = ackBit;
```

```java
    int ignoreBit = r.nextInt(2);
    if (ignoreBit > 0) {this.ackBit = !this.ackBit;} notify();
  }
  public synchronized boolean get() {
    while (!channelEmpty) {try {wait();} catch (
    InterruptedException e){}}
    channelEmpty = true; notify();  return protocolBit;
  }
  public synchronized void put(boolean protocolBit) {
    while (!channelEmpty) {try {wait();} catch (
    InterruptedException e){}}
    int ignoreBit = r.nextInt(2);
    channelEmpty = false; if (ignoreBit > 0) channelEmpty = true
    ;
    notify();
  }
}
class Sender implements Runnable {
  private LossyChannel channel;
  public Sender(LossyChannel channel) {this.channel = channel;}
  public void run() {
    boolean protocolBit = false; Random r = new Random();
    while (true) {
      if (protocolBit != channel.getAckBit()) channel.put(
    protocolBit);
      else protocolBit = !protocolBit;
      try {Thread.sleep(r.nextInt(1500));} catch (
    InterruptedException e){}
    }
  }
}
class Receiver implements Runnable {
  private LossyChannel channel;
  public Receiver(LossyChannel channel) {this.channel = channel
    ;}
  public void run() {
    Random r = new Random();
    while (true) {
      boolean protocolBit = channel.get(); channel.putAckBit(
    protocolBit);
      try {Thread.sleep(r.nextInt(1500));} catch (
    InterruptedException e){}
    }
  }
}
```

2.4 Zadanie 4 - napisz kod generujący poniższy kod

Dane są kolory jak na wykładzie. Zdefiniuj własny styl. aby wyglądał jak ten poniżej.

```java
import␣java.util.Random;
public␣class␣AltBitProtocol␣{
  public static void main(String[] args) {
    LossyChannel channel = new LossyChannel();
    (new Thread(new Sender(channel))).start();
    (new Thread(new Receiver(channel))).start();
  }
}
class LossyChannel {
  private boolean protocolBit, channelEmpty = true, ackBit = true;
  private Random r;
  public LossyChannel() {r = new Random();}
  public synchronized boolean getAckBit() {notify(); return ackBit;}
  public synchronized void putAckBit(boolean ackBit) {
    this.ackBit = ackBit;
    int ignoreBit = r.nextInt(2);
    if (ignoreBit > 0) {this.ackBit = !this.ackBit;} notify();
  }
  public synchronized boolean get() {
    while (!channelEmpty) {try {wait();} catch (InterruptedException e){}}
    channelEmpty = true; notify();  return protocolBit;
  }
  public synchronized void put(boolean protocolBit) {
    while (!channelEmpty) {try {wait();} catch (InterruptedException e){}}
    int ignoreBit = r.nextInt(2);
    channelEmpty = false; if (ignoreBit > 0) channelEmpty = true;
    notify();
  }
}
class Sender implements Runnable {
  private LossyChannel channel;
  public Sender(LossyChannel channel) {this.channel = channel;}
  public void run() {
    boolean protocolBit = false; Random r = new Random();
    while (true) {
      if (protocolBit != channel.getAckBit()) channel.put(protocolBit);
      else protocolBit = !protocolBit;
      try {Thread.sleep(r.nextInt(1500));} catch (InterruptedException e){}
    }
  }
}
class Receiver implements Runnable {
  private LossyChannel channel;
  public Receiver(LossyChannel channel) {this.channel = channel;}
  public void run() {
    Random r = new Random();
    while (true) {
```

```
␣␣␣␣␣␣boolean␣protocolBit␣=␣channel.get();␣channel.putAckBit(protocolBit);
␣␣␣␣␣␣try␣{Thread.sleep(r.nextInt(1500));}␣catch␣(InterruptedException␣e){}
␣␣␣␣}
␣␣}
}
```

2.5 Zadanie 5 - napisz kod generujący poniższy kod

Dane są kolory jak na wykładzie. Zdefiniuj własny styl. aby wyglądał jak ten poniżej.

```
import␣java.util.Random;
public␣class␣AltBitProtocol␣{
␣␣public␣static␣void␣main(String[]␣args)␣{
␣␣␣␣LossyChannel␣channel␣=␣new␣LossyChannel();
␣␣␣␣(new␣Thread(new␣Sender(channel))).start();
␣␣␣␣(new␣Thread(new␣Receiver(channel))).start();
␣␣}
}
class␣LossyChannel␣{
␣␣private␣boolean␣protocolBit,␣channelEmpty␣=␣true,␣ackBit␣=␣true;
␣␣private␣Random␣r;
␣␣public␣LossyChannel()␣{r␣=␣new␣Random();}
␣␣public␣synchronized␣boolean␣getAckBit()␣{notify();␣return␣ackBit;}
␣␣public␣synchronized␣void␣putAckBit(boolean␣ackBit)␣{
␣␣␣␣this.ackBit␣=␣ackBit;
␣␣␣␣int␣ignoreBit␣=␣r.nextInt(2);
␣␣␣␣if␣(ignoreBit␣>␣0)␣{this.ackBit␣=␣!this.ackBit;}␣notify();
␣␣}
␣␣public␣synchronized␣boolean␣get()␣{
␣␣␣␣while␣(!channelEmpty)␣{try␣{wait();}␣catch␣(InterruptedException␣e){}}
␣␣␣␣channelEmpty␣=␣true;␣notify();␣␣return␣protocolBit;
␣␣}
␣␣public␣synchronized␣void␣put(boolean␣protocolBit)␣{
␣␣␣␣while␣(!channelEmpty)␣{try␣{wait();}␣catch␣(InterruptedException␣e){}}
␣␣␣␣int␣ignoreBit␣=␣r.nextInt(2);
␣␣␣␣channelEmpty␣=␣false;␣if␣(ignoreBit␣>␣0)␣channelEmpty␣=␣true;
␣␣␣␣notify();
␣␣}
}
class␣Sender␣implements␣Runnable␣{
␣␣private␣LossyChannel␣channel;
␣␣public␣Sender(LossyChannel␣channel)␣{this.channel␣=␣channel;}
␣␣public␣void␣run()␣{
␣␣␣␣boolean␣protocolBit␣=␣false;␣Random␣r␣=␣new␣Random();
␣␣␣␣while␣(true)␣{
```

```
␣␣␣␣␣␣if␣(protocolBit␣!=␣channel.getAckBit())␣channel.put(
    protocolBit);
␣␣␣␣␣␣else␣protocolBit␣=␣!protocolBit;
␣␣␣␣␣␣try␣{Thread.sleep(r.nextInt(1500));}␣catch␣(
    InterruptedException␣e){}
␣␣␣␣}
␣␣}
}
class␣Receiver␣implements␣Runnable␣{
␣␣private␣LossyChannel␣channel;
␣␣public␣Receiver(LossyChannel␣channel)␣{this.channel␣=␣channel
    ;}
␣␣public␣void␣run()␣{
␣␣␣␣Random␣r␣=␣new␣Random();
␣␣␣␣while␣(true)␣{
␣␣␣␣␣␣boolean␣protocolBit␣=␣channel.get();␣channel.putAckBit(
    protocolBit);
␣␣␣␣␣␣try␣{Thread.sleep(r.nextInt(1500));}␣catch␣(
    InterruptedException␣e){}
␣␣␣␣}
␣␣}
}
```

Listing 1: Java example

3 Pakiet `algorithm2e`

Stosując pakiet `algorithm2e` rozwiąż poniższe zadania.

3.1 Zadanie 1 - napisz kod generujący poniższy algorytm

Z użyciem komendy `\DontPrintSemicolon`

```
Input: A finite set $A = \{a_1, a_2, \ldots, a_n\}$ of integers
Output: The largest element in the set
$max \leftarrow a_1$
for $i \leftarrow 2$ to $n$ do
    if $a_i > max$ then
        $max \leftarrow a_i$
    end
end
return $max$
```

Algorithm 1: Max finds the maximum number

Bez użycia komendy `\DontPrintSemicolon`

```
Input: A finite set $A = \{a_1, a_2, \ldots, a_n\}$ of integers
Output: The largest element in the set
$max \leftarrow a_1$;
for $i \leftarrow 2$ to $n$ do
    if $a_i > max$ then
        $max \leftarrow a_i$;
    end
end
return $max$;
```

Algorithm 2: MAX finds the maximum number

3.2 Zadanie 2 - napisz kod generujący poniższy algorytm

Algorithm 1 CHANGE Makes change using the smallest number of coins

```
Input: A set $C = \{c_1, c_2, \ldots, c_r\}$ of denominations of coins, where $c_i > c_2 > \ldots > c_r$ and a positive number $n$
Output: A list of coins $d_1, d_2, \ldots, d_k$, such that $\sum_{i=1}^{k} d_i = n$ and $k$ is minimized
$C \leftarrow \emptyset$  for $i \leftarrow 1$ to $r$ do
    while $n \geq c_i$ do
        $C \leftarrow C \cup \{c_i\}$  $n \leftarrow n - c_i$
    end
end
return $C$
```

3.3 Zadanie 3 - napisz kod generujący poniższy algorytm

Algorithm 2 FINDDUPLICATE

```
Input: A sequence of integers $\langle a_1, a_2, \ldots, a_n \rangle$
Output: The index of first location witht he same value as in a previous location in the sequence
$location \leftarrow 0$  $i \leftarrow 2$  while $i \leq n$ and $location = 0$ do
    $j \leftarrow 1$  while $j < i$ and $location = 0$ do
        if $a_i = a_j$ then
            $location \leftarrow i$
        else
            $j \leftarrow j + 1$
        end
    end
    $i \leftarrow i + 1$
end
return $location$
```

3.4 Zadanie 4 - napisz kod generujący poniższy algorytm

Algorithm 3 FindDuplicate2

```
Input: A sequence of integers $\langle a_1, a_2, \ldots, a_n \rangle$
Output: The index of first location witht he same value as in a previous
        location in the sequence
$location \leftarrow 0 \ \ i \leftarrow 2$
while $i \leq n \land location = 0$ do
    $j \leftarrow 1$  while $j < i \land location = 0$ do
        if $a_i = a_j$ then  $location \leftarrow i$ ;
        else  $j \leftarrow j + 1$ ;
    end
    $i \leftarrow i + 1$
end
return $location$
```