/

ICS-A1120 Programming 2

Exam May 25th, 2014

Petteri Kaski (050-430 0948) and Tommi Junttila (050-430 0861)

The use of electronic devises is not allowed in the exam.

1. What is the decimal (base 10) representation of the unsigned binary integer 01010101 (given in the most-significant-bit-first order)?

   What is the 8-bit unsigned binary representation (in the "most significant bit first" order) of the base 10 integer 151?

2. Consider an integer `x` whose type is `Int`. Give a Scala-expression that returns an integer with type `Int` such that the bit position 22 has value 1, and all other bit positions agree with the corresponding bit positions in `x`. (In an integer of type `Int`, the least significant bit is at position 0, the most significant bit is at position 31.)

   Reminder: The bitwise NOT of a word can be obtained with the operator `~`, the bitwise AND of two words can be obtained with the operator `&`, and the bitwise OR in turn with the operator `|`. The bit positions in a word can be shifted to the left (towards more significant bits) with the operator <<, and to the right (towards less significant bits) with the operators >> and >>>.

3. Construct a Boolean circuit according to the following specification, using the gates NOT (`!`), OR (`||`), and/or AND (`&&`). The circuit has two input elements `a`, `b`, and one output `r`. The input elements may be independently assigned either to the value `false` or to the value `true`. The value of the output `r` must be `true` exactly when the inputs `a` and `b` have the same value (either both `true` or both `false`). Present the circuit either as a Scala-expression or as a drawing, where the input elements, the output, and the types of the gates (NOT, OR, AND) have been clearly indicated.

4. Let us consider the following simple assembly-language program. What is the value stored in register $0 when the program stops (halt)?

```
 mov $0, 0       # move 0 to $0
 mov $1, 1       # move 1 to $1
@loop:
 cmp $1, 10      # compare $1 and 10
 bab >done       # branch to 'done' if > holds in most recent comparison
 add $0, $0, $1  # add $1 to $0
 add $1, $1, 2   # add 2 to $1
 jmp >loop       # jump to 'loop'
@done:
 hlt             # halt
```

5. Describe in one or two sentences what the function below (written in imperative programming style) computes. Give a corresponding function written in functional programming style.

```
def f(s: IndexedSeq[Int], p: Int => Boolean): Int = {
  var i = 0
  var r = 0
  while(i < s.length) {
    val v = s(i)
    if(p(v)) {
      r = r + v
    }
    i = i + 1
  }
  r
}
```

You may use the classes and methods in the `scala.collections.immutable` package.

Hint: the methods `filter` and `sum` may be useful.

6. For both functions given below, use big O notation to describe the growth rate of the run-time with respect to the length n of the argument sequence `s`. The growth rate functions you provide should have as slow growth as possible.

   Recall that indexed access and evaluating the `length` method are constant time operations in the `Array` type.

   Function 1:

```
def myFunc1(s: Array[Int]): Int = {
  require(s.nonEmpty)

  def myMax(): Int = {
    var r = s(0)
    var i = 1
    while(i < s.length) {
       if(s(i) > r)
         r = s(i)
       i = i + 1
    }
    r
  }

  var i = 0
  var freq = 0
  while(i < s.length) {
    if(s(i) == myMax())
      freq = freq + 1
    i = i + 1
  }
  freq
}
```

   Function 2:

```
def myFunc2(s: Array[Int]): Int = {
  require(s.nonEmpty)

  def myMax(): Int = {
    var r = s(0)
    var i = 1
    while(i < s.length) {
       if(s(i) > r)
         r = s(i)
       i = i + 1
    }
    r
  }

  var i = 0
  var freq = 0
  val v = myMax()
  while(i < s.length) {
    if(s(i) == v)
      freq = freq + 1
    i = i + 1
  }
  freq
}
```

7. In your own words (that is: without giving any program code), describe the idea of binary search.

   Assume that we use binary search to answer the question "does the array below contain the number 63?". Which elements in the array are compared to the number 63 during the binary search?

```
Array(2,3,4,5,21,29,31,34,35,35,37,500,1000)
```

8. Consider the function `f` given below. Describe in one sentence (with words, no code allowed) what the function computes. Illustrate the call stack when `f` is called with the parameter `l` set to the value `List(1,2,6)`.

```
def f(l: List[Int]): Double = {
  def inner(r: List[Int], s: Int, n: Int): (Int, Int) = {
    if(r.isEmpty) (s, n)
    else inner(r.tail, s + r.head, n + 1)
  }
  val (s, n) = inner(l, 0, 0)
  if(n == 0) 0.0
  else s.toDouble / n.toDouble
}
```

9. Let us recall the linked lists data structure presented in the lecture notes. Is the method `count` below in tail-recursive form? If not, explain why this is the case and also give a corresponding tail-recursive version in functional programming style.

```
abstract class LinkedList[A] {
  def isEmpty: Boolean
  def head: A
  def tail: LinkedList[A]
  def count(p: A => Boolean): Int = {
    this match {
      case Nil() => 0
      case Cons(h, t) =>
        if(p(h)) 1 + t.count(p)
        else t.count(p)
    }
  }
}
case class Nil[A]() extends LinkedList[A] {
  def isEmpty = true
  def head = throw new java.util.NoSuchElementException("head of empty list")
  def tail = throw new java.util.NoSuchElementException("tail of empty list")
}
case class Cons[A](val head: A, val tail: LinkedList[A]) extends
    LinkedList[A] {
  def isEmpty = false
}
```

10. Let `t` be an arbitrary string (`String`), for example

```
val t = "soossi"
```

Present a function (or a one-line expression) that returns an array (`Array`) of strings consisting of all the non-empty prefixes of a string `t` given as a parameter to the function. For the string given above the return value would thus be

```
val result = Array("s", "so", "soo", "soos", "sooss", "soossi")
```

Reminder: Each string has an accompanying method `take(k)` that returns a string consisting of the `k` first characters in the string.

11. Consider the execution of the multithreaded program below:

```
import scala.concurrent._
import ExecutionContext.Implicits.global

object mystery {
  def main(args: Array[String]) {
    for(i <- 0 until 2) {
      future { print("1") }
      future { print("2") }
      future { print("3") }
      future { print("4") }
    }
  }
}
```

Which of the following are possible outputs from the program?

(a) 11 (b) 12341234 (c) 43214321 (d) 11223344 (e) 22142134 (f) 33333333 (g) 12134243

12. Relative to lexicographic order, does it hold that

(a) "sakka" < "sushi", (b) "sanaluettelo" < "sana", (c) "324" < "2", (d) "345" < "3456" ?

Please answer either yes or no to each of (a,b,c,d).