**LECTURE 15 WORKSHEET**

Please, work in groups of **3 people**. Upload this worksheet at the end of the class.

**NAMES: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |
| --- |
| **PART 1. PSEUDO-RANDOM NUMBER GENERATORS** |

* 1. How do you think the Kahoot! name generator works? Please, describe your idea in plain English or using pseudocode

|  |
| --- |
|  |

* 1. Write down as many ideas as you can for generating randomness with a computer

|  |
| --- |
|  |

1.3. Using a Linear Congruential Generator (LCG), use the values a=11, c=0, m=8, X0=3 and complete the following table:

|  |  |
| --- | --- |
| *i* | *Xi* (random value) |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |

Considering your proposal for the Kahoot! name generator, what name alternatives would be offered to the user by this random number generator if the user wants to get a new name?

|  |
| --- |
|  |

In your opinion, is this a good random number generator? Why?

|  |
| --- |
|  |

What characteristics should a good number generator have?

|  |
| --- |
|  |

**1.4.** Find values for *a*, *c* and *m* that allow to achieve full period. Write them down here:

|  |
| --- |
|  |

Select a value for X0:

and calculate the first *m* random numbers of your generator. Write them down here:

|  |
| --- |
|  |

By simple visual inspection, what of the following points does this generator comply with?

|  |  |
| --- | --- |
| 1. Full period? | YES/ NO |
| 2. Evenly distributed in range [0,*m*-1]? | YES/ NO |
| 3. Independent numbers? \* | YES/ NO |
| 4. Fast? | YES/ NO |

\* *Roughly speaking, if the numbers show some pattern, they are not independent. Examples of patterns: all the numbers are increasing, decreasing, they alternate between being over a certain value and under that value, there is a relationship between every kth number, etc. There are “serious” tests for this, but if you spot a pattern, they are not independent (if you don´t, it does not mean they are independent).*

What do you think is better? a long period or a short period? Why?

|  |
| --- |
|  |

1.5. For a=29, c=35, m=256 (set of values meeting the conditions for full period) and X0=1, the first 10 pseudo-random numbers and their binary representation are shown in the leftmost columns:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **i** | **Xi (decimal)** | **Xi (binary)** | **Decimal b0** | **Decimal b1b0** | **Decimal b2 b1b0** |
| 1 | 64 | 01000000 |  |  |  |
| 2 | 99 | 01100011 |  |  |  |
| 3 | 90 | 01011010 |  |  |  |
| 4 | 85 | 01010101 |  |  |  |
| 5 | 196 | 11000100 |  |  |  |
| 6 | 87 | 01010111 |  |  |  |
| 7 | 254 | 11111110 |  |  |  |
| 8 | 233 | 11101001 |  |  |  |
| 9 | 136 | 10001000 |  |  |  |
| 10 | 139 | 10001011 |  |  |  |
| 11 | 93 | 01011101 |  |  |  |
| 12 | 172 | 10101100 |  |  |  |
| 13 | 159 | 10011111 |  |  |  |
| 14 | 38 | 00100110 |  |  |  |
| 15 | 113 | 01110001 |  |  |  |
| 16 | 240 | 11110000 |  |  |  |
| 17 | 83 | 01010011 |  |  |  |
| 18 | 138 | 10001010 |  |  |  |

Complete the table by writing down the equivalent decimal number if you just consider the last bit b0 (rightmost bit), the last 2 bits and the last 3 bits. For example, for the number 87 the decimal numbers for the last, 2 last and 3 last bits would be: 1, 3 and 7.

What is your conclusion regarding the behaviour of the low-order bits of this generator?

|  |
| --- |
|  |

**1.6 Xorshift**

Calculate the random number generated by xorshift(2,1,4,10). Assume an 8-bit word.

Number 1:

|  |
| --- |
|  |

If you enter the number just generated as the seed for another call of xorshift (with the same shift parameters, just a different seed) then you get the second number and so on. Calculate the next 2 random numbers.

Number 2:

Number 3:

|  |
| --- |
| **PART 2. COMPARISON SORTS** |

**2.1.** **BOGOSORT:** for the following array of numbers:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 | 2 | 9 | 3 | 5 |

[0] [1] [2] [3] [4]

Run **just one iteration of the loop** in Bogosort and write down the content of the array:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

Is it possible that Bogosort runs indefinitely? Why?

|  |
| --- |
|  |

**2.2. INSERTION SORT:** for the following array of numbers:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 | 2 | 9 | 3 | 5 |

[0] [1] [2] [3] [4]

Write down the content for the array immediately after finishing every for-loop iteration:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

Looking at the result obtained after every for-loop iteration, describe **in words** what INSERTION SORT does in every iteration:

|  |
| --- |
|  |

**2.3. MERGE (vector)**

Describe, in words, the purpose of lines 3-5:

|  |
| --- |
|  |

Describe, in words, the purpose of lines 7-13:

|  |
| --- |
|  |

If a=[2, 4, 8, 9] and b=[1,3,5,6], what is the content of array c after executing MERGE(a,b)?

|  |
| --- |
|  |

**2.4. MERGESORT**

For the array:

|  |  |  |  |
| --- | --- | --- | --- |
| 5 | 3 | 13 | 2 |

Draw the best diagram you can think of to depict the execution of mergesort, take a picture of it and paste it here.

|  |
| --- |
|  |

**2.5. QUICKSORT**

1. Write down 7 random numbers, each in a small piece of paper (so they fit in the small table you have in front of you). Shuffle them and put them in a line, from left to right in no particular order.
2. Select one number (for example, the rightmost one). This is the **pivot** (the pivot can be any element, not only the rightmost).
3. Your mission is: finding the right final position of the pivot, making sure all lower/higher numbers are to its left /right.
4. Complete the initial collection of numbers here:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |

Complete the content of the array after the first call to partition:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |

Complete the content of the array after the second call to partition:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |

Using the same numbers, what is the worst-case position of numbers? (that producing a maximum number of swaps during the execution of partition)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |

|  |
| --- |
| **PART 3. SHUFFLING** |

Given an array of numbers, write down one idea for an algorithm that reorders the numbers according to a (uniformly) random permutation.

|  |
| --- |
|  |