CSGames Qualifications 2020

Theoretical Computing

~~Use this Latex template to answer directly~~

Question 1

Name the data structure that fits the description:

1. Element search is in O(log n)

   Liste

2. Element insertion is in O(1)

   Hashset

3. Element lookup is in O(1)

   Liste

Question 2

Name a sorting algorithm that can complete in O(n)

Sleep sort FTW

Question 3

Can a sorting algorithm finish in O(log n) without preprocessing the data?

No it can't

Question 4

Explain what is a regular language?

Un language pouvant être accepté par un automate fini

Question 5

If we have v = {a, b}, give v*.

J'assume que c'est la fermeture de Kleene de v right? If not, I don't know.
Is so, c'est toute combinaison de a ou b incluant le mot vide.

Question 6

Given the regular expression R = (ab∗)∗, find an equivalent regular expression that only have a star height of 1 (no nested Kleene star). The allowed operations are union, concatenation and Kleene star.

R = (a (a U b)*) U ε

Question 7

Is this equation true or false? justify. $(\neg r \Rightarrow q) \vee \neg(q \vee s) \vee s \wedge q \wedge r \wedge \neg(t \wedge s) \Rightarrow (q \wedge \neg t) \vee (r \wedge \neg s)$

Question 8

If a language L is regular, it's complement is also a regular language. true or false? justify.

Vrai, comme L est régulier, il y a un automate qui accepte L. On peut inverser « l'acceptation » (acceptant devient rejetant et vice-versa) de tout les états de l'automate qui accepte L pour faire celui qui accepte le complément de L, donc le complément est un language régulier.

Question 9

Explain in details the impact of a black box that could factorize products of prime numbers in polynomial time on the rsa cryptosystem.

La seule chose qui rend RSA sécure est le fait qu'on ne peut pas factoriser de très grands nombres en un temps raisonnable. Le fait de pouvoir détruirait complètement le système.

Question 10

Explain the following regular expression: ε ∪ 0(0 ∪ 1)* ∪ 1((0((1(0 ∪ 1)(0 ∪ 1)*) ∪ 0(0 ∪ 1)*)*) ∪ (1(0 ∪ 1)(0 ∪ 1)*))*

3 parties:

ε -> Le mot vide

0(0 U 1)* -> Ou tout mot commençant par 0

1 ((0 [(0 U 1)(0 U 1)*] U [1(0 U 1)(0 U 1)*])* -> Sinon, un mot commençant par 1 et, s'il y a plus d'une lettre, il faut qu'il soit suivi d'un 0 et d'une autre lettre. Si la lettre qui suit le 0 est 1, il doit y avoir une autre lettre après.

Question 11

Evaluate Ω = (λx.xx)(λx.xx) 2

No goddam idea, pas la matière Q.Q

Question 12

Using Church encoding, evaluate λx.λy.x(xy)

Same here Q.Q

Question 13

Explain, in your own words, Turing's proof of undecidability of the halting problem.

Je sais que c'est un preuve comme quoi on ne saura jamais si un programme halte, mais aucune idée c'est quoi sa preuve.

Question 14

Prove that L = {XX : X ∈ P? } is an irregular language.

En utilisant le lemme de pompage.

Prenons un mot de longueur 2p.
Le lemme affirme que, pour tout p et n, il existe un x, y, z ou w = x(y^n)z et |xy| = p
De plus, |y| > 0.

Or, avec un mot de longueur 2p, on a xy = z = X
Avec un n égal à quoique ce soit d'autre que 1, il est très clair que xy != z, donc L n'est pas régulier.

Question 15

Prove P = NP

No puedo