Nom	
Prénom	N
Groupe	El

Note \(\begin{aligned} \begin

Algorithmique Contrôle 1 - Partie 1 INFO-SUP S1 EPITA 31 Oct. 2022 - 08:30	
 Ceci est la partie 1 de l'épreuve - Vous devez rendre les deux parties! Vous devez répondre directement sur ce sujet. Répondez dans les espaces prévus, les réponses en dehors ne seront pas corrigées. Aucune réponse au crayon de papier ne sera corrigée. La présentation est notée. 	
Exercice 1 (Un peu de cours – 4 points)	
1. Quel type de données retourne un observateur? M observateur retourne un argument de type prédéting of	V
2. Comment appelle-t-on une opération qui n'est pas définie partout? We opération partielle que	
3. Quelles sont les deux façons de définir une liste? - récursaive v sont les deux façons de définir me - cténative V liste.	
4. Quelles zones constituent la signature d'un type abstrait?	
Les zones qui constituent la signature d'un type abstract sont: OPERATIONS, TYPES et UTILISE.	
5. Donnez trois opérations définissant une liste récursive (sans places). - cons: élément x liste -> liste - fin: liste-> liste et tête: liste-> élément	

325

Nom	PELOU
Prénom	Lucile
Groupe	El
Prof TD	

Note My /16

$\begin{array}{c} {\rm Algorithmique} \\ {\rm Contr\^{o}le} \ 1 \ \text{- Partie} \ 2 \end{array}$

Info-sup S1 Epita

31 Oct. 2022 - 08:30

Remarques (à lire!):

Ceci est la partie 2 de l'épreuve - Vous devez rendre les deux parties!	
Vous devez répondre directement sur ce sujet. — Répondez dans les espaces prévus, les réponses en dehors ne seront pas corrigées. — Aucune réponse au crayon de papier ou au stylo rouge ne sera corrigée.	
 CAML: Tout code CAML non indenté ne sera pas corrigé. En l'absence d'indication dans l'énoncé, les seules fonctions que vous pouvez utiliser sont finvalid_arg (aucune autre fonction prédéfinie de CAML). Tout code CAML doit être suivi du résultat son évaluation: la réponse de CAML. 	ailwith et
La présentation est notée.	

Exercice 2 (Insertion après – 3,5 points)

Écrire la fonction insert_post x f 1st qui prend en paramètres :

- un élément x
- une fonction à un paramètre f renvoyant une valeur booléenne
- une liste 1st

et qui insère l'élément x dans la liste 1st juste après le premier élément y tel que f y est true.

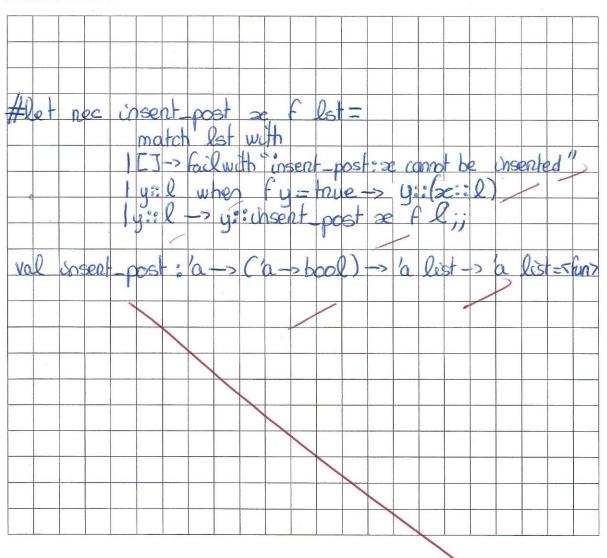
La fonction devra déclencher une exception Failure s'il n'existe aucun élément y après lequel x peut être inséré.

```
# insert_post 42 (function y -> y mod 2 = 0) [5; -7; 2; 4; 1];;
- : int list = [5; -7; 2; 42; 4; 1]

# insert_post "toto" (function y -> y = "one") ["one"; "two"; "three"; "four"];;
- : string list = ["one"; "toto"; "two"; "three"; "four"]

# insert_post 4.1 (function y -> y < 0.) [1.1; 2.4; 2.];;
Exception: Failure "insert_post: x cannot be inserted".</pre>
```

Fonction CAML:



Exercice 3 (Position du maximum - 4,5 points)

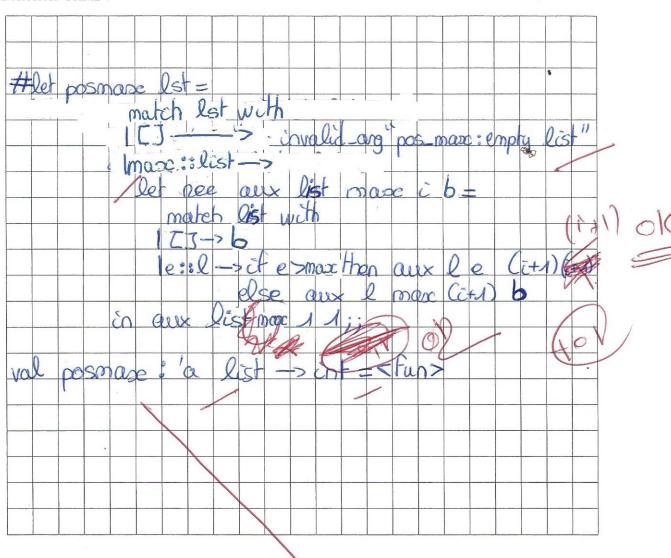
4

Écrire la fonction posmax 1st qui retourne la position de l'élément maximum de la liste 1st. On suppose que la liste 1st ne contient aucun doublon et la position du premier élément de la liste est 1.

La fonction devra déclencher une exception Invalid_argument si la liste 1st est initialement vide.

```
# pos_max [];;
Exception: Invalid_argument "pos_max: empty list".
# pos_max [1; 2; 3];;
- : int = 3
# pos_max [3; 2; 1];;
- : int = 1
# pos_max [8.5; 9.; -4.5];;
- : int = 2
```

Fonction CAML:



Exercice 4 (Less - 5 points)

1. Écrire la fonction less2 p k l1 12 dont les spécifications sont les suivantes :

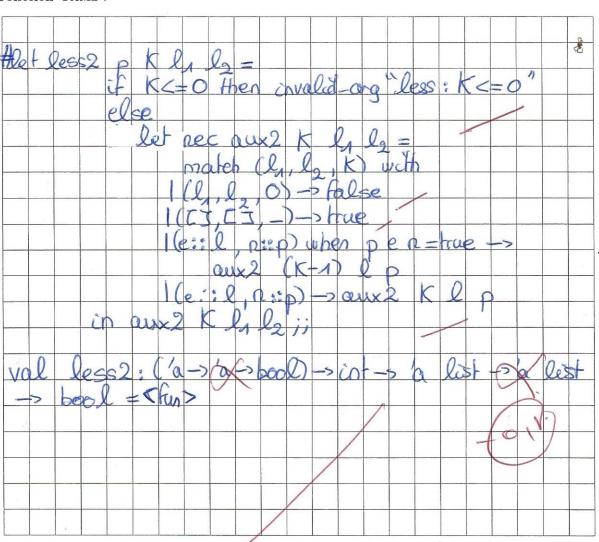
- Elle prend en paramètre une fonction à deux paramètres p, un entier strictement positif k ainsi que deux listes : $[a_1; a_2; ...; a_n]$ et $[b_1; b_2; ...; b_n]$. Les deux listes sont supposées de même longueur.
- Elle retourne true si le nombre de paires d'éléments (a_i, b_i) tels que p a_i b_i est true est strictement plus petit que k, et false sinon.
- Elle déclenche une exception Invalid_argument si le paramètre k est invalide.
- # less2 (function x -> function y -> x = y) 2 [] [];;
 : bool = true

 # less2 (function x -> function y -> x = y) 0 [5; 1; 1; 2] [1; 5; 1; 4];;
 Exception: Invalid_argument "less: k <= 0".

 # less2 (function x -> function y -> x > y) 3 [5.; 1.1; 1.8; 2.5] [1.; 5.7; 1.9; 5.];;
 : bool = true

 # less2 (function x -> function y -> x mod y = 0) 2 [5; 15; 5; 2] [7; 5; 4; 4];;
 : bool = true

Fonction CAML:



2. Utiliser la fonction précédente less2 (qu'elle soit écrite ou pas) pour écrire la fonction common k 11 12 qui vérifie si les listes 11 et 12 ont strictement plus de k éléments de mêmes valeurs aux mêmes positions. Les deux listes sont supposées de même longueur et le paramètre k est supposé positif ou nul.

```
# common 0 ['a'; 'y'; 'c'] ['c'; 'a'; 'y'];;
- : bool = false
# common 2 [1; 2; 3; 4] [1; 2; 3; 5];;
- : bool = true
# common 1 ['a'; 'y'; 'c'] ['b'; 'y'; 'c'];;
- : bool = true
```

Fonction CAML:



Exercice 5 (Mystery - 3 points)

1. Donner le type de la fonction suivante.

```
# let mystery1 l1 l2 =
   let rec aux l3 = function
      (l1, [])     -> [l1]
   | (l1, 0::12)     -> l3::aux [] (l1, l2)
   | (e::l1, i::l2)     -> aux (e::l3) (l1, (i-1)::l2)
   in
      aux [] (l1, l2);;
```

val mystery 1: intlist -> int list -> intlist list =< Fun>

2. On suppose la fonction mystery1 ci-dessus correcte et dans l'environnement Caml. Donner les résultats des évaluations successives des phrases suivantes si elles sont correctes. Si elles sont incorrectes indiquez "Erreur".

```
# mystery1 [1; 2; 3; 4; 5; 6; 7; 8; 9; 10] [4; 1; 2];;
```

-: int list list = [[4;3;2;1]; [5]; [7;6]; [8;9;10]]

mystery1 [1; 2; 3; 4; 5; 6; 7; 8; 9; 10] [7; 3];;

-:int list list = [[4;6;5;4;3;2;1];[10;9;8];[]]

3. Donner une application de la fonction mystery1 renvoyant le résultat ci-dessous.

```
# mystery1 ???;;
- : int list list = [[5; 15; 5]; [21]; [5; 1; 4]]
```

mystery/ [5;15;5;21;5;1;4] [3;1];;