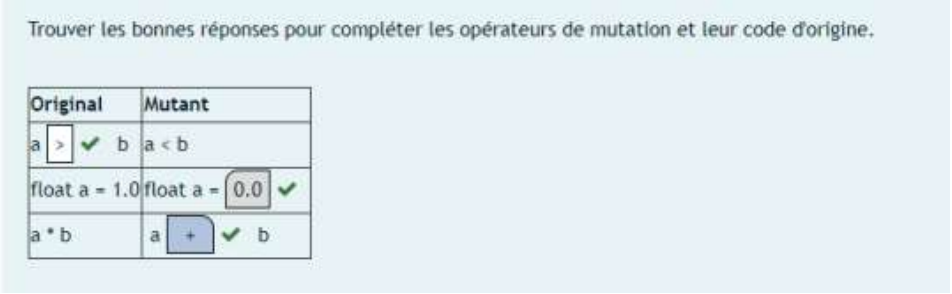
1. Que signifie une couverture d'instruction de 100%?
   R : Toutes les instructions de code ont été exécutées par la suite de tests

2. Un des objectifs des tests de mutation est de simuler des fautes classiques (e.g., faute de frappe).
   R : Vrai

3. Trouver les bonnes réponses pour compléter les opérateurs de mutation et leur code d'origine.
   R : Original → a>b Mutant→float a = 0.0 ainsi que → a+b

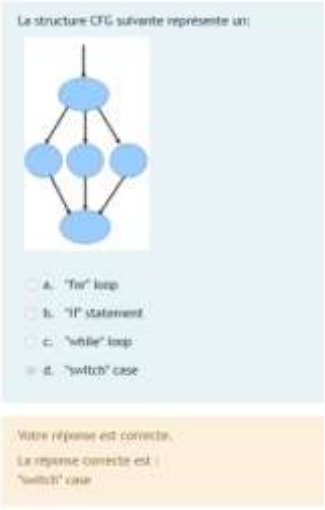


4. Ce sont toujours les gros bogues qui entraînent des échecs les plus coûteux.
   R : Faux

5. Sur la base du code suivant, trouvez la/les lignes de définitions et lignes d'utilisation des variables :

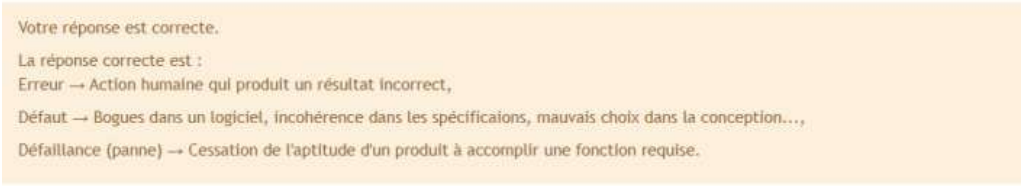
   R :

6. La structure CFG suivante représente un :



   R : switch case

7. Associez les termes à leurs définitions/exemples.



8. Les oracles de test sont souvent écrit à la main par les développeurs.
   R : Vrai

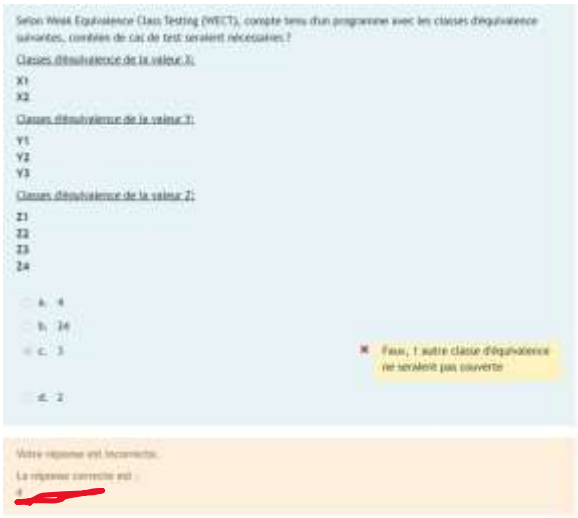
9. Les tests boite noire permettent de trouver plus de fautes que les tests boite blanche
   R : Faux

10. L'analyse des valeurs limites est bien adaptée pour tester un programme qui prend en entrée des chaînes de caractères, car elle nous permettra de révéler des fautes de conception.
    R : Faux

11. Pour la méthode de Catégorie-Partition, un paramètre peut avoir :
    R : Toutes ces réponses

12. Choisissez toutes les réponses qui s'appliquent.
    Les tests boîte noire :
    R : Sont basés sur les exigences d'un programme
        Considèrent les entrées et les sorties d'un programme

13. Selon Weak Equivalence Class Testing (WECT), compte tenu d'un programme avec les classes d'équivalence suivantes, combien de cas de test seraient nécessaires?
    R : 4



14. Considérez le fonctionnement (simplifiée) d'une imprimante.

La réponse correcte est :
Considérez le fonctionnement (simplifiée) d'une imprimante.

(Astuce, le seul moment où l'imprimante ne fait rien, c'est lorsqu'elle n'est pas alimentée.)

Remplissez le tableau de décision suivant. Utilisez le symbole "-" pour indiquer aucune action.

		Regles			
Conditions	Alimenté	[Oui]	Non	[Oui]	Non
	Demande d'impression reçue	Oui	Oui	[Non]	Non
Actions	Afficher l'état "On"	[X]	[-]	[X]	[-]
	Imprimer	[X]	[-]	[-]	[-]
	Rien faire	[-]	[X]	[-]	[X]

15. La forme normale disjonctive (DNF) est composé de :
    R : clauses conjonctives reliés par l'opérateur OU

16. Les cas suivants couvrent la couverture des clauses et des prédicats de Z = B~C + ABD :
    R : Faux

17. Il n'est pas toujours possible d'utiliser RACC, nous avons donc besoin de CACC pour couvrir des cas particuliers avec un critère moins rigoureux mais qui permet la couverture des prédicats et des clauses.

Il [n'est pas toujours] ✔ possible d'utiliser [RACC] ✔ , nous avons donc besoin de [CACC] ✔ pour couvrir des cas particuliers avec un critère moins rigoureux mais qui permet la couverture des prédicats et des clauses.

n'est jamais | est toujours

18. Une de vos amies développeur conçoit un nouveau site Web. Elle vous demande conseil pour structurer certaines options, elle envisage deux designs. Elle peut soit avoir 5 paramètres d'entré avec 3 options chacun, soit 3 paramètres d'entré avec 5 options chacun. Dans les deux cas, elle soupçonne des interactions entre 3 paramètres. Considérant uniquement les problèmes liés au test d'interactions entre les paramètres, quelle design suggéreriez-vous?
    R : 5 paramètres d'entré avec 3 options chacun

19. Selon le tableau ci-dessous :
    Z= A~B+C
    Choisissez tous les éléments qui sont des PPF (pour au moins une clause, n'importe quelle clause) :

    R : Cas 3, Cas 1

Selon le tableau ci-dessous:

Z= A~B+C

cas	A	B	C	A~B	Z
1	Faux	Faux	Faux	Faux	Faux
2	Faux	Faux	Vrai	Faux	Vrai
3	Faux	Vrai	Faux	Faux	Faux
4	Faux	Vrai	Vrai	Faux	Vrai
5	Vrai	Faux	Faux	Vrai	Vrai
6	Vrai	Faux	Vrai	Vrai	Vrai
7	Vrai	Vrai	Faux	Vrai	Vrai
8	Vrai	Vrai	Vrai	Vrai	Vrai

Choisissez tous les éléments qui sont des PPF (pour au moins une clause, n'importe quelle clause):

a. Cas #1
b. Cas #7
c. Cas #3
d. Cas #8

Votre réponse est correcte.

Les réponses correctes sont :
Cas #3,
Cas #1

20. Il est possible d'utiliser des pré-conditions et des post-conditions pour créer automatiquement des tests pour un logiciel.
    R : vrai

21. Soit la classe Ensemble" qui peut contenir un nombre indéterminé d'éléments sans doublons. Les attributs sont les éléments, et le compte du nombre d'éléments (compte). Les méthodes de la classe et leurs pré et post conditions sont les suivantes :

22. Créer un test pour qu'il soit complétement indépendant des autres sans utiliser d'autres méthodes pour faciliter le test (exemple, méthodes « helper »), de cadriciel, ou d'ensembles de données prédéfinies, c'est :
    R : Laborieux

23. Dans les tests de logiciels, un Stub est :
    R : Une simulation de comportement externe au composant que l'on souhaite tester

24. Faire correspondre les difficultés d'échafaudage à leurs solutions.
    R :
    Boucle infinie → Définir un délai acceptable pour le temps d'attente des réponses
    Logiciel avec une interface graphique → Remplacer la composante par un Driver/Stub pour tester le reste du système
    Logiciel avec des messages concurrents non-déterministe → Comparer une distribution attendue des résultats à la distribution réelle des résultats

25. Sélectionnez les éléments qui correspondent aux programmes orientés objet et comment nous les testons.
    R :
    Les sorties dépendent de l'état interne des classes
    La complexité des logiciels OO est dans les interactions entre méthodes

26. L'approche MaDUM est une approche boîte blanche qui nous permet de déterminer des tranches qui peuvent trouver n'importe quelles failles dans un programme orienté objet.
    R : Faux

27. Selon le principe de Liskov, une classe dérivée doit avoir toutes les méthodes de la classe de base, mais les signatures des méthodes de la classe dérivée ne doivent pas nécessairement être compatibles avec celles de la classe de base.
    R : Faux

28. Qu'est-ce qu'un Pilote de logiciel?
    R : Un composant qui appelle l'« Unité testée »

29. Qu'est-ce qu'un Talon de logiciel?
    R : Un composant dont dépend l'« Unité testée »

30. Ordonner les phases de test selon le processus traditionnel de gestion des tests.
    R :

31. Sélectionnez les cas qui sont des types de tests applicables au génie logiciel.
    R :
    Test d'exactitude
    Test de Fiabilité
    Test de Sécurité
    Test de performance
    Test de portabilité

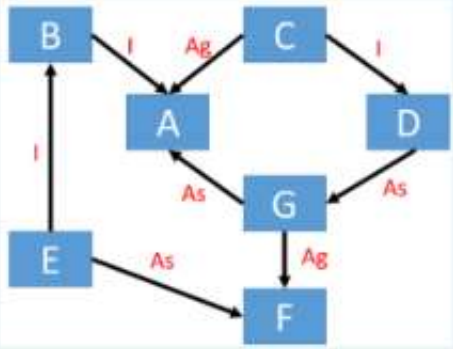
32. Lors de la création de systèmes ML, nous remplaçons principalement les éléments centrés sur l'humain (par exemple, le code) par des données. Cela a un impact sur la façon dont nous testons les systèmes. Vrai ou Faux?
    R : Vrai

33. Laquelle de ces propriétés suivantes n'est pas une propriété métamorphique pour les images?
    R : Changer la couleur d'un seul pixel de manière aléatoire

34. L'approche Big Bang est une combinaison des approches d'intégration descendante et ascendante.
    R : Faux

35. Soit l'image suivante :
    Pour chaque classe, déterminez l'Ordre partiel de tests (plusieurs classes peuvent appartenir à un même numéro d'ordre)
    R :

Soit l'image suivante :

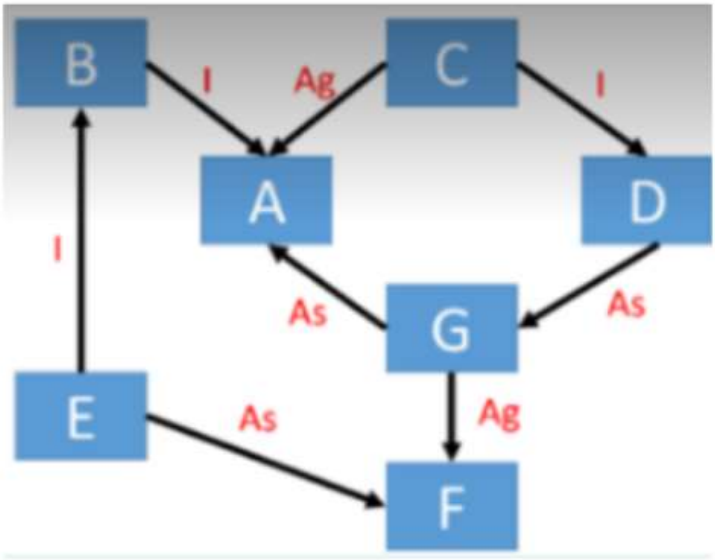


Soit l'image suivante :

Pour chaque classe, déterminez l'Ordre partiel de tests (plusieurs classes peuvent appartenir à un même numéro d'ordre)

Classe : Ordre

Classe A: 4 ✖
Classe B: 2 ✔
Classe C: 1 ✖
Classe D: 2 ✖
Classe E: 1 ✖
Classe F: 4 ✖
Classe G: 3 ✖



ordre partiel	classe
1	A, F
2	B, G
3	D, E
4	C

36. Pour tout programme, il est généralement possible de trouvé tous les tests qui seraient nécessaires pour s'assurer que notre programme n'a pas de défauts.
    R : Faux

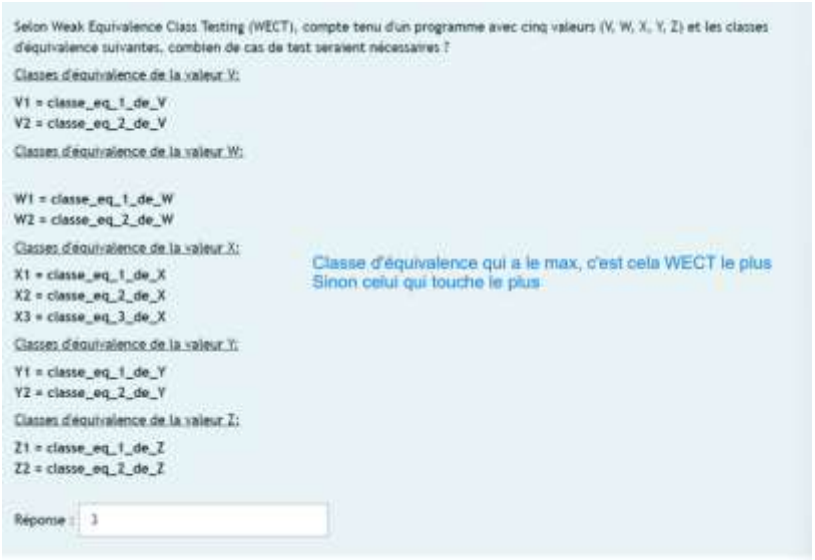
37. Les activités de test ne sont pas nécessaires pour des systèmes de petite taille (moins de 1000 LOC) développés en Java.
    R : Faux

38. Si nous avons la possibilité d'exécuter des tests sur chaque partie d'un logiciel séparément d'une manière exhaustive, nous n'aurons pas besoin de tester le système une fois qu'il sera assemblé.
    R : Faux

39. En utilisant le programme suivant
    En utilisant le code ci-dessus, complétez les DC-PATH suivants :

40. Pour le problème ci-dessus, quelle réponse ne contient pas un DU-Path pouvant être utilisé pour satisfaire le critère all-DEF?
    R : DU-PATH(txt, 4, 8)

41. Selon Weak Equivalence Class Testing (WECT), compte tenu d'un programme avec cinq valeurs (V,W,X,Y,Z) et les classes d'équivalence suivantes, combien de cas de test seraient nécessaires?
    R : 3



42. L'analyse des valeurs limites est une technique de test de la boîte **noire**.

L'analyse des valeurs limites fonctionne bien quand les variables ont une plage de valeurs **limitées**.
Elle est donc, **mal adaptée** pour tester un programme qui prend en entrée des chaînes de caractères.

L'analyse des valeurs limites est une technique de test de la boîte noire .

L'analyse des valeurs limites fonctionne bien quand les variables ont une plage de valeurs limitées .

Elle est donc, mal adaptée pour tester un programme qui prend en entrée des chaînes de caractères.

43. La fonction sin(x) reçoit en entrée un nombre réel (int) entre -180 et 180 degrés et retourne une valeur entre -1.0 et 1.0 ou ERREUR.
Complétez les classes d'équivalence suivantes.

44. La fonction cercle (p, ray, c) reçoit en entrée

C:

	Choix	Contraintes
C1	ROUGE	[si négatif & visible]
C2	JAUNE	[si négatif & visible] ✔
C3	BLEU	[si positif & visible] ✔
C4	BLANC	[si positif & visible]
C5	VERT	[si mix & visible]
C6	NON VALIDE	[error] ✔

S :

	Choix	Contraintes
S1	0<= S <=1	
S2	S <0	[error]
S3	S >1	[error]
S4	NON-Réel	[error]
S5	NON-Numérique	[error]

Trames de tests pour **critère Each Choice**:

P	R	C	S
P1	R1 ✔	C3	S1
P1	R1	C4 ✔	S1
P2	R1 ✔	C1	S1 ✔
P2	R1 ✔	C2 ✔	S1
P3 ✔	R1 ✔	C5	S1
P4	R1 ✔	C5	S1

45. Nous pouvons dire que nous faisons ici des tests en boîte grise (pas sûr !! Voir contexte question) car nous mélangeons boîte noire (RWCT) et boîte blanche (test unitaire avec code source).

46. Si nous avons accès au programme correspondant.
    R : Il est généralement impossible de déterminer si un jeu de tests est idéal

47. Il faut attendre qu'un code de programme soit complet pour commencer les activités de test.
    R : Faux

48. En utilisant le programme suivant
    Marquez chaque cellule si la cellule correspond à un DEF, P-USE ou C-USE.

En utilisant le programme suivant

```
1   public static void search(String txt, String pat)
2   {
3       int M = pat.length();
4       int N = txt.length();
5       for (int i = 0; i <= N - M; i++) {
6           int j;
7           for (j = 0; j < M; j++)
8               if (txt.charAt(i + j) != pat.charAt(j))
9                   break;
10          if (j == M)
11              System.out.println("Pattern found at index " + i);
12      }
13  }
```

Marquez chaque cellule si la cellule correspond à un DEF, P-USE ou C-USE. Les marqueur 'X' ne doivent pas toucher au lignes (doivent être à l'intérieur des cellules voulues). Certaines cellules sont déjà remplies (par exemple, "-" "X"), vous n'avez pas besoin de marquer ces cellules.

	Variable																	
	txt			pat			M			N			i			j		
Ligne	DEF	C-USE	P-USE	DEF	C-USE	P-USE	DEF	C-USE	P-USE	DEF	C-USE	P-USE	DEF	C-USE	P-USE	DEF	C-USE	P-USE
1	X			X														
2																		
3																		
4																		
5								X	-		X	-						
6																		
7																		
8		-	X		-	X												
9																		
10																		
11																		

49. Pour le problème ci-dessus, quelle réponse ne contient pas un DU-Path pouvant être utilisé pour satisfaire le critère all-DEF?
    R : DU-PATH(txt, 4, 8)

50. La fonction log(x) reçoit en entrée un nombre flottant réel positif et retourne le logarithme de ce nombre ou ERREUR. Complétez les classes d'équivalence suivantes. (Indice : quelle est la réponse à log(0.0)?)
    R :

Votre réponse est partiellement correcte.

Vous en avez sélectionné correctement 1.

EC1, 2, et 3 doivent couvrir tous les cas possible pour notre fonction. Il y a trois possibilités: entrée trop petite, entrée correcte, entrée trop grande.

x n'est pas un nombre flottant réel (Erreur)
x <= 0.0 sont tous des erreurs avec des nombres valides (float)
x > 0.0 sont tous des nombres valides

La réponse correcte est :
La fonction log(x) reçoit en entrée un nombre flottant réel positif et retourne le logarithme de ce nombre ou ERREUR. Complétez les classes d'équivalence suivantes. *(Indice : quelle est la réponse à log(0.0)?)*

Classes d'équivalence de l'entrée

EC1 : x [n'est pas un nombre flottant réel]

EC2 : x [<=] 0.0

EC3 : x [>] 0.0

51. Pour la fonction logique Z = A(BC + ~D). En utilisant les méthodes de couverture des conditions logiques générez les jeux de tests pour la méthode RACC :

Pour la fonction logique **Z = A(BC + ~D)**. En utilisant les méthodes de couverture des conditions logiques générez les jeux de tests pour la méthode RACC :

La table de vérité est fournie ci-dessous.

Variant	A	B	C	D	Z
0	0	0	0	0	F
1	0	0	0	1	F
2	0	0	1	0	F
3	0	0	1	1	F
4	0	1	0	0	F
5	0	1	0	1	F
6	0	1	1	0	F
7	0	1	1	1	F
8	1	0	0	0	T
9	1	0	0	1	F
10	1	0	1	0	T
11	1	0	1	1	F
12	1	1	0	0	T
13	1	1	0	1	F
14	1	1	1	0	T
15	1	1	1	1	T

RACC:

A: 0 et 8 ✔

B: 11 et 15 ✔

C: 13 et 15 ✔

D: 8 et 9 ✔

52. Considérez le fonctionnement (simplifiée) d'une imprimante.

Votre réponse est partiellement correcte.

Vous en avez sélectionné correctement 14.

La réponse correcte est :

Considérez le fonctionnement (simplifiée) d'une imprimante.

(Astuce, le seul moment où l'imprimante ne fait rien, c'est lorsqu'elle n'est pas alimentée.)

Remplissez le tableau de décision suivant. Utilisez le symbole "-" pour indiquer aucune action.

		Regles			
Conditions	Alimenté	[Oui]	Non	[Oui]	Non
	Demande d'impression reçue	Oui	Oui	[Non]	Non
Actions	Afficher l'état "On"	[X]	[-]	[X]	[-]
	Imprimer	[X]	[-]	[-]	[-]
	Rien faire	[-]	[X]	[-]	[X]

53. Le coupe-feu de classe (CFW) pour une classe Y est l'ensemble des classes qui sont dépendante de la classe Y. Le CFW représente les classes qui devraient être testées quand la classe Y est changée.

Votre réponse est correcte.

La réponse correcte est :

Le coupe-feu de classe (CFW) pour une classe Y est [l'ensemble des] classes qui [sont dépendentante] de la classe Y.

Le CFW représente les classes qui [devraient être testées] quand la classe Y est changée.

54. Vous êtes chargé de choisir une stratégie de publication pour le système. Une équipe d'assurance qualité (QA) a déjà proposé quatre stratégies différentes (illustrées ci-dessous).

Temps restant 0:00:44

Question 20
Réponse enregistrée
Noté sur 1,00
Marquer la question

Vous êtes chargé de choisir une stratégie de publication pour le système. Une équipe d'assurance qualité (QA) a déjà proposé quatre stratégies différentes (illustrées ci-dessous).

Stratégies	Facteur de forme k	Facteur d'échelle λ
Stratégie 1 Nous ne retardons pas la sortie. Nous testons en utilisant notre équipe de test actuelle	2.0	20
Stratégie 2 Nous retardons la sortie de 4 semaines. Nous doublons le nombre de ressources dont nous disposons pour les tests.	1.5	30
Stratégie 3 Nous retardons la sortie de 4 semaines. Nous conservons les ressources actuelles de notre équipe de test.	1.5	20
Stratégie 4 : Nous devançons la sortie de 4 semaines. Nous augmentons les ressources de test par un facteur de 5.	0.8	1

L'objectif de l'entreprise est d'avoir moins de 10% de chance d'un défaut dans les 31 premiers jours. L'équipe QA indique également que l'ajout de 5 membres supplémentaires aux équipes de test actuelles triplerait toutes les valeurs de Facteur d'échelle (λ) pour toutes les stratégies ci-dessus sans affecter les valeurs de Facteur de forme (k).

Quelle stratégie choisiriez-vous ? Souhaitez-vous ajouter 5 membres supplémentaires ? Expliquez votre raisonnement.

La formule utilisée pour calculer la distribution cumulative sera F(x) = 1-e^[-(x/λ)^k], où x est le nombre de jours

Stratégie 1

λ=20 et k=2.0 => F(31) = 91%

Stratégie 2

55. L'entreprise émergente souhaite produire plusieurs composants de son système en parallèle, mais elle aimerait également produire un prototype le plus tôt possible. Ils planifient actuellement les composants suivants :
ORD

Temps restant 0:00:56

Question 19
Réponse enregistrée
Noté sur 1,50
Marquer la question

L'entreprise émergente souhaite produire plusieurs composants de son système en parallèle, mais elle aimerait également produire un prototype le plus tôt possible. Ils planifient actuellement les composants suivants :

- Une interface utilisateur Android qui permet aux utilisateurs de soumettre des clips audios et de voir si l'audio de ces clips existe dans un film ou une émission de télévision.
- Un composant pour enregistrer et sauvegarder des clips audios soumis par les utilisateurs.
- Un composant qui prend la sortie d'un composant d'apprentissage automatique et la transforme en chaînes de caractères pouvant être transmises à l'interface utilisateur.
- Un composant d'apprentissage automatique qui prend en entrée les clips audios soumis par l'utilisateur, et des représentations audio d'émissions de télé et de films. Le composant a pour sortie dans quelle émission de télévision ou quel film le clip existe (le cas échéant).
- Un composant pour transformer des émissions de télévision et des films obtenus à partir de bases de données en représentations audio.
- Un composant pour accéder à des bases de données de films
- Un composant pour accéder à des bases de données d'émissions de télévision

Cette question comporte **deux parties:**

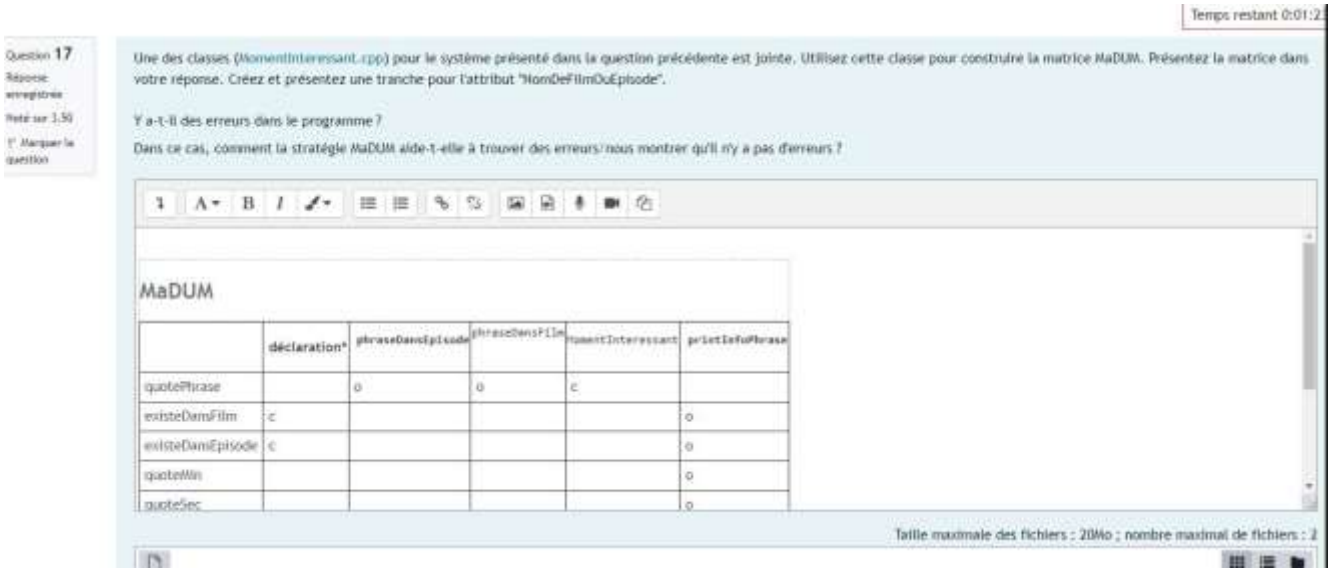
1) Vous êtes chargé d'identifier une stratégie de test d'intégration. Quelle stratégie proposez-vous et pourquoi (1-4 phrases) ?

2) Créez un diagramme de relation d'objet (ORD) pour les composants proposés et utilisez la stratégie de Kung et al pour déterminer un ordre de test partiel. Soumettez votre diagramme et votre ordre de test partiel. Écrivez toutes les hypothèses que vous formulez dans le cadre de cette question.

Hypothèse: L'utilisateur peut enregistrer un audio sans qu'il (l'audio) soit soumis à l'analyse.

1) Tout d'abord Big bang ne peut pas être considéré vu la complexité de l'application. De plus, malgré que la stratégie descendante nous donnerait un prototype rapidement, mais on n'a pas une grande connaissance des algorithme qui vont être utilisé au bas de l'arbre (principalement comment la construction du modèle). La stratégie ascendante serait idéal si l'équipe possédait plus de temps, mais ce n'est pas le cas. En combinant les deux approches, l'idéal serait d'intégrer avec la stratégie de sandwich modifié puisqu'on aura un composant puisqu'un

56. Une des classes (MomentInteressant.cpp) pour le système présenté dans la question précédente est jointe. Utilisez cette classe pour construire la matrice MaDum. Présentez la matrice dans votre réponse. Créez et présentez une tranche pour l'attribut « NomDeFilmOuEpisode ».



57. L'entreprise émergente souhaite que ses utilisateurs puissent enregistrer de l'audio sur leur appareil Android et utiliser cet audio pour extraire un fichier audio afin de lancer le processus de recherche.