Société
informatique
de France

Analyse et propositions relatives au programme de terminale de spécialité Numérique et Sciences Informatiques

27 juin 2019

Commentaire général

Le programme des deux années de la spécialité « Numérique et sciences informatiques », première et terminale, couvre un large spectre de la science informatique. En cela, il devrait enfin permettre aux lycéens de se construire une représentation plus juste de cette science multifacette et de ses applications, en montrant concrètement que l'informatique ne se résume pas à la programmation.

La part importante laissée aux projets (1/4 du temps en première, 1/3 en terminale) peut être une source de motivation pour les lycéens. Elle peut également leur donner un aperçu des vastes champs d'application de l'informatique et permettre de jouer l'interdisciplinarité que la réforme du lycée et la grande combinatoire des choix de spécialités rendent plus que difficiles à mettre en œuvre. La variété des sujets proposés ouvre de belles perspectives quels que soient les autres choix de spécialité des lycéens. On peut juste s'étonner qu'au milieu des différents thèmes évoqués, un seul algorithme (l'algorithme A* pour la recherche d'itinéraires sur une carte) soit explicitement cité. Donner des pistes d'algorithmes utilisables par les lycéens pour résoudre tel ou tel problème serait très utile pour l'ensemble des thèmes proposés et trouverait parfaitement sa place dans un document d'accompagnement des programmes. Concernant les projets, les élèves seront sans doute amenés à utiliser des outils logiciels ou des bibliothèques tierces, voire des ressources multimédias. Il est indispensable qu'ils s'interrogent sur la qualité des ressources choisies mais aussi sur les licences d'exploitation qui leur sont associées, et éventuellement sur les données à fournir pour utiliser la ressource. C'est d'autant plus vrai, s'ils sont amenés à utiliser des services web où le devenir des données qui transitent doit être pris en compte.

Concernant l'articulation du programme en première puis terminale, une progression logique est construite tout au long des deux années. Les notions de représentation de données vues en première s'étendent à l'étude de structures de données en terminale, le traitement de données en table se poursuit logiquement par une introduction aux bases de données. Les questions liées au matériel sont également abordées en première puis approfondies en terminale. Bien entendu, les langages de programmation et l'algorithmique occupent aussi une part importante dans les deux programmes. Seuls les aspects « interactions entre l'homme et la machine sur le web » travaillés en première ne trouvent pas explicitement leur pendant dans le programme de terminale.

Cependant, après étude, notamment en collaboration avec les formateurs du DIU Enseigner l'Informatique au Lycée et les enseignants en cours de formation, le programme de NSI paraît, au regard de la maturité des lycéens et du temps dévolu à la spécialité, trop ambitieux. Nous proposons donc tout d'abord des pistes pour alléger certaines parties du programme mais aussi renforcer certains apprentissages pour favoriser leur assimilation (Allégements et renforcements proposés). Nous proposons ensuite de préciser ou développer certains attendus afin de mieux cerner ce qui doit être acquis par les élèves et évalués à l'issue de la formation (Suggestions de précisions et d'explicitations des savoirs). Puis, nous nous interrogeons sur deux aspects connexes au programme. Nous posons tout d'abord quelques questions relatives aux liens entre NSI et les autres disciplines (Croisements avec les autres disciplines). Puis, nous nous interrogeons sur les modalités de mise en œuvre du programme (Modalités de mise en œuvre). En annexe, nous proposons une très rapide comparaison en termes de volume horaire de travail entre les deux années de spécialité au lycée et les deux premières années de licence informatique qui justifie en partie nos demandes d'allègement (Annexe I : spécialité NSI vs programmes d'informatique en L1/L2). Nous présentons très brièvement comment se déclinent les attendus en termes de savoirs et de savoir-faire selon les différents thèmes dans les programmes de première et terminale (Annexe II : nature des attendus des différents thèmes présents dans les programmes de première et terminale). Enfin nous récapitulons de manière synthétique les modifications proposées pour le programme (Annexe III : tableau récapitulatif des modifications proposées dans le programme de terminale.) ainsi que les éléments qu'il nous paraitrait pertinent d'insérer dans des documents d'accompagnement (Annexe IV : préconisation sur les documents d'accompagnement).

Allégements et renforcements proposés

Très concrètement, voici une liste des capacités attendues dans le programme de Terminale qui nous paraissent dépasser ce que l'on peut légitimement espérer d'un lycéen dans le cadre de la spécialité. Ce sont des capacités qui sont actuellement transmises dans les cursus universitaires au mieux à partir de la 3ème année de licence informatique voire en master informatique.

- Structures de données : il serait peut-être préférable de se concentrer sur les structures linéaires (listes, piles, files), les dictionnaires et les arbres et se contenter, concernant les graphes, d'amener les élèves à découvrir que certains problèmes se modélisent à l'aide de graphes sans aller jusqu'à leur implémentation. Nous proposons donc d'enlever :
    - Écrire les implémentations correspondantes d'un graphe : matrice d'adjacence, liste de successeurs / prédécesseurs
    - Passer d'une représentation à une autre
- Langages et programmation : parmi les capacités citées dans cette rubrique, certaines demandent des facultés d'abstraction et d'analyse et un certain recul sur la discipline probablement. Vu le volume horaire alloué, il ne nous paraît pas réaliste d'attendre qu'un lycéen soit capable de
    - Montrer sans formalisme théorique que le problème de l'arrêt est indécidable,
    - Choisir le paradigme de programmation selon le champ d'application d'un programme

  De manière générale, il ne nous semble pas raisonnable de mettre explicitement au programme « calculabilité et décidabilité »

- Algorithmique : dans la partie « structures de données », il est question des structures de listes, piles, files, dictionnaires et on attend d'un lycéen qu'il soit capable de choisir la structure la plus adaptée au problème à résoudre. Il serait sans doute donc logique de voir apparaitre dans la section algorithmique des algorithmes qui utilisent explicitement ces structures de données. Par ailleurs, là encore certaines capacités, semblent difficile à exiger d'un lycéen :
    - « Rechercher une clé dans un arbre de recherche, insérer une clé » : il faudrait, *a minima*, préciser que l'insertion se fait au niveau des feuilles. On peut également se demander si les arbres binaires de recherche ne pourraient éventuellement pas plutôt apparaître en commentaires comme exemple d'arbres que dans la partie capacités attendues.
    - « Algorithmes sur les graphes » : comme pour leur implémentation, il nous semble que ces algorithmes sont hors de portée dans le contexte du lycée. Il est sans doute possible de donner certaines intuitions notamment via l'application au routage, ou via des manipulations débranchées mais il nous semble que la construction rigoureuse de ces algorithmes dès la terminale posera des difficultés. Nous proposons donc d'ôter les éléments suivants des capacités attendues :
        - « Parcourir un graphe en profondeur d'abord, en largeur d'abord.
        - Repérer la présence d'un cycle dans un graphe
        - Chercher un chemin dans un graphe »
    - « Recherche textuelle » : « Étudier l'algorithme de Boyer-Moore pour la recherche d'un motif dans un texte » : là encore cet algorithme ne nous parait pas accessible à la plupart des lycéens, il serait sans doute plus adapté de les inciter à réfléchir à des algorithmes plus simples de manipulation de chaînes de caractères (qui ne sont, pour l'heure, mentionnés explicitement ni dans le programme de Première ni dans celui de Terminale). On peut, par exemple, étudier les algorithmes (récursifs et/ou itératifs) qui permettent de dire si une chaîne de caractères est un palindrome, une anagramme, etc. On peut aussi étudier les algorithmes qui disent si une expression mathématique, représentée par une chaîne de caractères, est correctement parenthésée. On peut aussi demander à écrire un algorithme qui permet d'évaluer une expression arithmétique post fixée simple ce qui met également en œuvre la structure de pile.

      Dans la partie Algorithmique, il nous semble également que certains exemples donnés en commentaires, même s'ils ne sont pas prescriptifs, ne sont pas les plus adaptés. Ainsi, renseignement pris auprès de plusieurs enseignants-chercheurs, dont certains spécialisés en algorithmes de manipulations d'images, « la rotation d'une image bitmap d'un quart de tour avec un coût en mémoire constant », outre une certaine difficulté, n'est pas une méthode standard de manipulation d'images et ne fait pas non plus partie des exemples classiques utilisés pour illustrer les méthodes de type « diviser pour régner ». L'« alignement de séquences » s'il est, quant à lui, très répandu dans les enseignements de bio-informatique demande lui aussi un certain recul et n'apparait généralement qu'en 3ème année de licence. Pour l'approche diviser

pour régner, on pourrait suggérer dans les commentaires de revenir sur le tri dichotomique vu dans le programme de première. De même pour la programmation dynamique, on pourrait, outre le rendu de monnaie, étudier à nouveau le problème du sac à dos qui seraient ainsi résolus en terminale avec une autre approche que celle utilisée en Première. L'intérêt pédagogique est immédiat et illustre bien qu'un problème peut être résolu de différentes manières.

- Architectures matérielles, système d'exploitation et réseaux : l'écart est important entre les programmes de Première et de Terminale.
    - « Composants intégrés d'un système sur puce » : Pour pouvoir évoquer les systèmes sur puce, sans imposer de compétences attendues, il serait préférable d'aborder en premier les circuits séquentiels. Le contenu précédemment cité pourrait être remplacé par « Circuits combinatoires et séquentiels, architecture d'une machine ». Les capacités attendues pourraient être « Identifier sur le schéma d'une machine, la partie opérative, la partie contrôle, les mémoires et les chemins de données. Décrire le fonctionnement d'un programme simple en langage machine. ». On pourrait ajouter en commentaires : « On prolonge le travail entrepris en classe de première sur les circuits combinatoires et le modèle de machine de Von Neumann en introduisant quelques circuits séquentiels simples (compteurs, mémoires). »
    - « Protocoles de routage » : l'explicitation détaillée des algorithmes de routage RIP ou OSPF semble hors de portée des lycéens de terminale et de leurs enseignants. On propose donc de remplacer le cadre "Protocoles de routage." par « Réseau », avec comme capacités attendues : « Comprendre les différents niveaux de protocoles : physique, liaison, réseau, transport, application. Identifier l'importance des algorithmes de routage dynamique. » et en commentaires : « On prolonge le travail entrepris en classe de première sur les protocoles et l'architecture des réseaux en abordant la structure du réseau mondial et en distinguant les réseaux publics et privés. »

Suggestions de précisions et d'explicitations des savoirs

Les capacités attendues dans les programmes de Première et Terminale NSI sont essentiellement de deux sortes (reconnaissables via les verbes employés) : des capacités liées à un savoir, avec plus ou moins de recul exigé (« distinguer », « identifier », « décrire », « comprendre », « reconnaître », « analyser » etc.), d'autres reposant sur des savoir-faire (« écrire », « utiliser », « construire », « réaliser », etc.).

Certains thèmes visent de manière assez équilibrée l'obtention de capacités des deux sortes, d'autres en revanche sont très fortement voire presque exclusivement axés sur des savoirs.
Que les lycéens acquièrent une culture générale en informatique est important mais si cette culture ne s'incarne pas dans des savoir-faire, elle risque de rester lettre morte voire de donner une image erronée de ce qui sera attendu ultérieurement lors de leurs études supérieures. Il semblerait que cela ait été le cas avec de précédents programmes, en physique notamment.

D’autre part, les capacités relevant des savoir-faire sont bien circonscrites. Mais des précisions sur certains attendus relevant plutôt du savoir, voire la déclinaison de certains en savoir-faire seraient utiles, pour aider les enseignants à construire leurs séquences pédagogiques et leurs évaluations. Certaines de ces précisions seraient bienvenues dans le programme, d’autres pourraient être seulement données dans un document d’accompagnement.

- Histoire de l’informatique : s’il est essentiel de replacer l’informatique dans son histoire, les quelques repères temporels présents dans les programmes de Première et Terminale ne permettent pas d’asseoir l’ensemble de l’enseignement NSI sur un corpus commun d’informations historiques. Des repères historiques plus nombreux pourraient être proposés dans les documents d’accompagnement.
- Structures de données : ici des savoir-faire sont systématiquement associés à des savoirs, il n’est pas besoin d’expliciter davantage. Si la préconisation sur l’ajout d’algorithmes sur les listes, pile, files etc. est suivie, on pourrait ajouter dans les commentaires, la phrase « On fait le lien avec la rubrique « algorithmique ». D’autre part, on peut signaler une petite coquille dans les commentaires associés justement aux piles et files. Il faudrait Remplacer : "On distingue les modes FIFO et LIFO des piles et des files" par : "On distingue les modes FIFO et LIFO des files et des piles".
- Bases de données : ici une grande partie des capacités attendues est de l’ordre du savoir. Il pourrait être opportun de remplacer la capacité "Identifier les concepts définissant le modèle relationnel" par une capacité plus opérationnelle : "Savoir, sur un exemple, décomposer une relation et identifier les clés possibles" et ainsi rééquilibrer savoir et savoir-faire. De même, on pourrait remplacer « identifier les composants d’une requête » par « être capable d’expliquer la structure d’une requête »
- Architecture matérielles, systèmes d’exploitation : En ce qui concerne la « gestion des processus et des ressources par un système d’exploitation », l’expression « Décrire la création d’un processus, l’ordonnancement de plusieurs processus par le système » est un peu vague, peut-être vaudrait-il mieux là encore aller vers une capacité plus opérationnelle : « Savoir définir ce qu’est un processus, identifier les processus actifs sur une machine et comprendre la nécessité de l'ordonnancement des processus par le système. ». Dans les commentaires, on pourrait aussi indiquer qu’une présentation débranchée de quelques algorithmes d’ordonnancement peut être réalisée. Ces algorithmes utilisent des piles et des files et sont donc également une illustration des structures de données vues dans le programme. Enfin, pour se placer dans la continuité du programme de Première, on pourrait ajouter en commentaires « les élèves utilisent un système d’exploitation libre »
- Langage et programmation : ici nous avons préconisé le retrait de la plupart des capacités de type savoir qui demandait un recul trop important sur la discipline. Par ailleurs, il nous semble important de donner des précisions sur des savoir-faire mentionné autour de la « modularité ». Il serait en effet pertinent d’insister ici en commentaire sur le fait que l’utilisation d’API ou de bibliothèques doit se faire de manière éclairée, notamment sur la qualité des fonctionnalités proposées (complexité des opérations par exemple). Concernant Python, langage support choisi en première et terminale, il est clairement dit en préambule des deux programmes que « l’expertise dans tel ou tel langage de programmation n’est […] pas un objectif de formation ». Or Python dispose d’un grand nombre de raccourcis d’écriture qui lui sont propres,

peuvent rendre le code écrit peu lisible (ce qui en informatique est un réel problème) et ne se retrouvent pas tel quel dans d'autres langages de programmation. Il serait bon dans un document d'accompagnement de préciser les limites d'utilisation des « facilités » de python. Le sujet 0 de QCM pour les élèves de première qui arrêtent la spécialité avant d'entrer en terminale est de ce point de vue assez emblématique de ce qu'il ne faudrait pas exiger des élèves de lycée.

D'autre part, contrairement à d'autres langages plus stricts voire rigoureux, l'interpréteur de Python n'aide pas le programmeur débutant à éviter certains bugs (pas d'exigence de typage explicite, non détection d'instructions conditionnelles non exhaustives, etc.). Pour pallier ce problème, il serait utile de fournir dans les documents d'accompagnement des règles de bonne pratique de programmation ainsi que recommander des outils qui aident à la recherche de bugs dans les applications.

- Algorithmique : Nous avons ici une interrogation par rapport aux proposition relatives aux épreuves pour le baccalauréat. En dehors des algorithmes de parcours des arbres explicitement évoqués, de ceux liés aux graphes et à la recherche textuelle actuellement proposés que nous proposons d'enlever et qui ne rentrent certainement pas dans la catégorie des algorithmes « ne présentant pas de difficulté particulière », aucun autre algorithme ne « figur[e] explicitement au programme ». Ce sont des familles d'algorithmes qui paraissent dans les capacités attendues pas des algorithmes précis. Ainsi, nous nous demandons sur quoi pourrait porter le premier exercice de la sous-épreuve pratique envisagée.

Croisements avec les autres disciplines

Comme indiqué en préambule, les projets peuvent être source de travaux pluridisciplinaires, cependant, l'étanchéité entre disciplines à laquelle sont soumis les programmes nous semblent poser un certain nombre de problèmes pour l'acquisition du contenu disciplinaire lui-même.

Par exemple, l'absence de mathématiques dans le tronc commun et l'absence de contrainte de suivre un enseignement de mathématiques lorsque l'on choisit la spécialité NSI impliquent que l'on ne peut pas supposer que les élèves ont connaissance d'un certain nombre d'éléments, pourtant nécessaires à l'enseignement du contenu du programme :

- La preuve par récurrence,
- La fonction logarithme voire la fonction puissance et leurs propriétés,
- Les suites,
- etc.

Doit-on en déduire qu'il reviendra au professeur de NSI d'enseigner lui-même ces compléments indispensables de mathématiques ?

De manière connexe, nous regrettons également qu'au sein de la spécialité maths ou du tronc commun de maths de seconde, les maths "en lien avec l'informatique", par exemple telles que proposées dans la contribution à l'élaboration d'un programme de maths par les 4 sociétés savantes de mathématiques et d'informatique en 2016[1] - n'aient pas leur place.

[1] https://www.societe-informatique-de-france.fr/2016/10/propositions-maths-info-lycee/

Modalités de mise en œuvre

Il est indiqué dans cette partie du programme que « les activités pratiques et la réalisation de projets supposent que chaque élève ait un accès individuel à un équipement relié à Internet ». Il nous semble que cette condition si elle est nécessaire est loin d'être suffisante. Les échanges avec les collègues qui vont enseigner NSI montrent une grande disparité dans les configurations matérielles et logicielles dont ils vont disposer. Il nous semble que des préconisations nationales sur un panel de logiciels et des configurations minimales de machines utiles pour enseigner NSI pourraient avantageusement accompagner la mise en place de cet enseignement.

Nous souhaitons aussi attirer l'attention sur les conditions humaines de mise en place de cet enseignement. Les collègues qui vont assurer cet enseignement en première dès septembre prochain sont pour la plupart des collègues ayant déjà enseigné l'informatique dans le cadre plus restreint de la spécialité ISN. Ils suivent actuellement une formation complémentaire lourde via un diplôme universitaire pour acquérir les compétences nécessaires à ce nouvel enseignement. Mais il ne faut pas oublier que lors de la première année de mise en place de la spécialité, un grand nombre d'entre eux, devra, outre ce nouvel enseignement à préparer, continuer à enseigner ISN en terminale, poursuivre et valider la formation universitaire, préparer les cours de sa propre discipline suite aux changements de programme et parfois intervenir ou épauler les collègues intervenant en Sciences Numériques et Technologie en seconde. Nous tenons à saluer l'investissement et le courage de ces enseignants et soulignons l'importance de tout mettre en œuvre pour les accompagner au mieux dans cette mission particulièrement complexe.

Annexe I : spécialité NSI vs programmes d'informatique en L1/L2

En ce qui concerne le contenu scientifique des programmes, il serait intéressant de le comparer à celui de curricula construits dans d'autres pays. Pour l'heure, les préconisations proposées ici se fondent essentiellement sur l'observation conjointe et la comparaison avec les contenus des deux premières années de licence dont il semble se rapprocher. Les échanges avec les professeurs du secondaire en cours de formation et leur connaissance des élèves de lycée permettent d'atténuer un des biais dont souffre cette comparaison. En effet, les étudiants qui sont actuellement en licence informatique ont une maturité plus grande que les lycéens et de fait, deux ans supplémentaires de culture scientifique au sens large. D'autre part, il faut aussi tenir compte d'une disparité importante concernant le volume horaire disciplinaire dans les deux premières années de licence informatique, même si celui-ci varie d'une formation à l'autre. Sans avoir mené une recherche exhaustive, il semble que le nombre de crédits européens (ECTS) actuellement alloués à la part disciplinaire en L1 et L2 informatique soit compris dans une fourchette allant de 55 à 65 ECTS. Concrètement, cela correspond, là encore avec des disparités selon les formations, à environ 550 à 650h de présentiel pour les étudiants auquel s'ajoute le travail personnel attendu[2] alors que l'on estime à 200h de travail (hors projet) le temps consacré à NSI en première et terminale.

[2] Pour mémoire, le travail global théoriquement attendu pour 1 ECTS est évalué à 25h minimum (travail présentiel inclus, cf https://publications.europa.eu/en/publication-detail/-/publication/da7467e6-8450-11e5-b8b7-01aa75ed71a1/language-fr). On pourrait donc

Annexe II : nature des attendus des différents thèmes présents dans les programmes de première et terminale

En première, les capacités attendues relèvent presque exclusivement de savoir-faire pour les représentations des données, le traitement de données en table, les langages de programmation et l'algorithmique. Elles sont presque uniquement de type savoir pour les interactions entre l'homme et la machine sur le Web et se répartissent à peu près équitablement selon les deux sortes dans la partie architectures matérielles et systèmes d'exploitation. En terminale, seule la partie algorithmique mentionne presque exclusivement des savoir-faire. Structures de données et langages de programmation affichent équitablement des capacités des deux familles, avec une exigence de recul pouvant être importante sur les aspects savoirs. Les parties bases de données et architectures matérielles, systèmes d'exploitation et réseaux mettent, quant à elles, l'accent sur les capacités de type savoir.

Il nous semble que d'un point de vue de l'apprentissage, les thèmes qui développent à la fois des savoirs et des savoir-faire seront ceux qui seront assimilés de manière plus durable par les lycéens.

Annexe III : tableau récapitulatif des modifications proposées dans le programme de terminale.

Seules apparaissent dans cette synthèse les modifications proposées, tout le reste nous parait convenir en l'état.

Dans la description de la démarche projet, nous proposons d'ôter la référence à l'algorithme A*.

Structures de données :

Contenus	Capacités attendues	Commentaires
Liste, piles, files : structures linéaires. Dictionnaires, index et clé	idem	On distingue les modes FIFO (*first in first out)* et LIFO (*last in first out)* **des files et des piles**

imaginer que le temps passé à travailler la discipline en L1 et L2 serait compris entre 1375h et 1625h, ce qui est dans les faits assez peu réaliste compte-tenu de la durée effective des années universitaires.

		On fait le lien avec la rubrique « algorithmique »
Graphes : structures relationnelles. Sommets, arcs, arêtes, graphes orientés ou non orientés	Modéliser des situations sous forme de graphe ~~Écrire les implémentations..~~ ~~Passer d'une représentation à une autre~~	On s'appuie sur des exemples comme le réseau routier, le réseau électrique, internet et les réseaux sociaux. **On pourra illustrer des manipulations sur les graphes en mode débranché.** ~~Le choix de la représentation… « algorithmique »~~

Bases de données :

Contenus	Capacités attendues	Commentaires
Modèle relationnel : relation, attribut, domaine, clef primaire, clef étrangère, schéma relationnel	**Savoir, sur un exemple, décomposer une relation et identifier les clés possibles**	idem
…		
Langage SQL : requêtes d'interrogation et de mise à jour d'une base de données	**Être capable d'expliquer la structure d'une requête.** Construire des requêtes ... INSERT, DELETE.	idem

Architectures matérielles, systèmes d'exploitation et réseaux

Contenus	Capacités attendues	Commentaires

~~Composants intégrés d'un système sur puce~~ **Circuits combinatoires et séquentiels, architecture d'une machine**	**Identifier sur le schéma d'une machine, la partie opérative, la partie contrôle, les mémoires et les chemins de données.** **Décrire le fonctionnement d'un programme simple en langage machine.**	**On prolonge le travail entrepris en classe de première sur les circuits combinatoires et le modèle de machine de Von Neumann en introduisant quelques circuits séquentiels simples (compteurs, mémoires).** **La notion de composants intégrés d'un système sur puce peut être abordée en complément.**
Gestion des processus et des ressources par un système d'exploitation	**Savoir définir ce qu'est un processus, identifier les processus actifs sur une machine et comprendre la nécessité de l'ordonnancement des processus par le système.** Mettre en évidence le risque de l'interblocage (*deadlock*)	À l'aide d'outils standard, il s'agit d'observer les processus actifs ou en attente sur une machine. Une présentation débranchée de l'interblocage **et de quelques algorithmes d'ordonnancement** peut être proposée. **Les élèves utilisent un système d'exploitation libre**
~~Protocoles de routage~~ **Réseau**	**Comprendre les différents niveaux de protocoles : physique, liaison, réseau, transport, application.** **Identifier l'importance des algorithmes de routage dynamique.**	**On prolonge le travail entrepris en classe de première sur les protocoles et l'architecture des réseaux en abordant la structure du réseau mondial et en distinguant les réseaux publics et privés.**

Langages de programmation

Contenus	Capacités attendues	Commentaires

Notion de programme en tant que donnée. ~~Calculabilité, décidabilité~~	Comprendre que tout programme est aussi une donnée. ~~Comprendre que la calculabilité… utilisé~~ ~~Montrer sans formalisme théorique, que le problème de l'arrêt est indécidable.~~ **Savoir expliquer la différence entre un interpréteur et un compilateur.**	L'utilisation d'un interpréteur… programme. **On peut faire écrire un programme qui écrit un programme.**
…		
Modularité	Idem	**L'utilisation d'API ou de bibliothèques doit se faire de manière éclairée notamment en termes de qualité des fonctionnalités proposées (complexité des opérations par exemple).**
Paradigmes de programmation	Distinguer sur des exemples… et objet. ~~Choisir le paradigme de programmation selon le champ d'application d'un programme~~	idem

Algorithmique

Contenus	Capacités attendues	Commentaires
Algorithmes itératifs ou récursifs utilisant des structures linéaires (listes, piles, files, chaînes de caractères etc.)	**Écrire des algorithmes utilisant ces structures**	**On peut s'intéresser aux algorithmes déterminant si chaîne de caractères est un palindrome, une anagramme.**

		On peut également travailler les algorithmes qui déterminent si une expression mathématique, représentée par une chaîne de caractères, est correctement parenthésée et à ceux qui permettent d'évaluer une expression arithmétique post fixée simple
Algorithmes sur les arbres binaires ~~et sur les arbres binaires de recherche.~~	Calculer… arbre Parcourir… (d'abord). ~~Rechercher une clé dans un arbre de recherche, insérer une clé au niveau des feuilles~~	Une structure de données récursive adaptée est utilisée. L'exemple des arbres permet d'illustrer la programmation par classe. ~~La recherche dans un arbre de recherche équilibré est de coût logarithmique.~~ **On peut éventuellement faire découvrir aux élèves les arbres binaires de recherche, comment y rechercher une clé, ou y insérer une clé au niveau des feuilles**
~~Algorithmes sur les graphes~~	~~Parcourir…d'abord.~~ ~~Repérer… graphe~~ ~~Chercher un chemin dans un graphe.~~	~~Le parcours d'un labyrinthe… programmation.~~
Méthode « diviser pour régner »	Idem.	**C'est l'occasion de revenir sur la recherche**

		dichotomique vue en classe de première. L'exemple du tri fusion… dans les pires des cas.
Programmation dynamique	Idem.	**Les exemples du problème du sac à dos ou** du rendu de monnaie **déjà rencontrés avec une autre approche en première peuvent être présentés.** La discussion sur le coût en mémoire peut être développée.
~~Recherche textuelle~~	~~Étudier… dans un texte~~	~~L'intérêt… exigée~~

Annexe IV : préconisation sur les documents d'accompagnement

Nous préconisons l'écriture d'un ou plusieurs documents d'accompagnement contenant, entre autres :

- des aspects historiques saillants en lien avec les différentes parties du programme et permettant d'illustrer la diversité des profils des personnes qui « font » la science informatique, on pourra notamment s'appuyer sur le jeu des 7 familles de l'informatique (https://interstices.info/dossier/7-familles-de-linformatique/) et l'enrichir d'autres personnalités.
- un guide de bonnes pratiques de programmation en python de manière générale mais également dans le cadre scolaire, afin que les connaissances acquises par les élèves soient facilement transposables lors de leurs études futures,
- des pointeurs vers des ressources pédagogiques permettant d'éclairer divers points du programme (relatifs aux savoirs et aux savoir-faire),
- des développements et des références autour des thèmes de projets évoqués dans le programme. Pour décliner les divers thèmes proposés en idées très concrètes de projets, on pourra s'inspirer du travail réalisé dans l'académie de Montpellier au sujet des projets d'ISN (http://www.ac-montpellier.fr/cid102250/des-idees-de-projets-pour-la-classe-en-isn.html) .
- des recommandations en terme d'environnement de travail avec une liste de logiciels et de configurations matérielles propices à l'enseignement de NSI.

Nous sommes tout disposés à contribuer sur ces points si besoin.