

<p>Langages objets</p> <p>Objets et fonctions (bloc 6)</p> <p>M2 Pro CCI, Informatique Emmanuel Waller, LRI, Orsay</p>	<p>résumé des épisodes précédents</p> <ul style="list-style-type: none"> • découverte Java, prise en main environnement, types primitifs • opérateurs et expressions, instructions de contrôle, débogage • tableaux • fonctions • objets (sans fonctions)
<p>objets et fonctions</p> <ul style="list-style-type: none"> • principe • exemple • répartition des fonctions dans les classes • Modification d'objets paramètres de fonctions • allocation dynamique d'objets dans fonctions 	<p>principe</p> <ul style="list-style-type: none"> • On maîtrise déjà les fonctions sur les tableaux • Avec objets : absolument rien de nouveau • On va simplement écrire des fonctions qui manipulent des objets (comme pour int, tableaux, etc.)
<p>exemple</p> <ul style="list-style-type: none"> • Rappel : reprenons les rationnels : créer et afficher (bloc Objets : Ex7.java) 	<p>exemple : avec fonctions</p> <ul style="list-style-type: none"> • On veut passer dans fonctions tout ce qui est possible • On va évidemment faire une fonction pour afficher, puis l'appeler : on va reprendre le même code et le placer dans une fonction • On verra le reste ensuite
<p>exemple</p> <ul style="list-style-type: none"> • Rationnels : fonction d'affichage (Ex1.java) 	<p>exemple : avec fonctions, revisité</p> <ul style="list-style-type: none"> • On peut aussi la placer dans la classe Rationnel • Dans ce cas, à l'appel, il faudra indiquer dans quelle classe se trouve la fonction qu'on appelle (comme pour les fonctions de la classe Math, et (presque) comme print et println dans la classe System) <ul style="list-style-type: none"> – Inutile si c'est la classe dans laquelle a lieu l'appel • C'est mieux <ul style="list-style-type: none"> – En fait, affichage dans Ex1 c'est mal

<div data-bbox="298 62 443 107" data-label="Section-Header"> <h2>exemple</h2> </div> <div data-bbox="60 154 564 226" data-label="List-Group"> <ul style="list-style-type: none"> • Rationnels : fonction d'affichage : le bon (Ex2.java) </div>	<div data-bbox="1093 62 1240 107" data-label="Section-Header"> <h2>exemple</h2> </div> <div data-bbox="858 154 1257 190" data-label="List-Group"> <ul style="list-style-type: none"> • Rationnels : complet (Ex3.java) </div>
<div data-bbox="218 638 520 683" data-label="Section-Header"> <h2>objets et fonctions</h2> </div> <div data-bbox="65 719 632 947" data-label="List-Group"> <ul style="list-style-type: none"> • principe • exemple • répartition des fonctions dans les classes • Modification d'objets paramètres de fonctions • allocation dynamique d'objets dans fonctions </div>	<div data-bbox="853 589 1479 728" data-label="Section-Header"> <h2>Répartition des fonctions dans les classes : dans quelle classe placer une fonction ?</h2> </div> <div data-bbox="858 748 1402 1046" data-label="List-Group"> <ul style="list-style-type: none"> • Sur les rationnels : dans la classe Rationnel • Sur les personnes : dans la classe Personne • Autre : dans la classe principale (celle qui contient main) • Possible, mais confus : <ul style="list-style-type: none"> – Fonction sur Rationnel dans classe principale – CCI : interdit </div>
<div data-bbox="90 1196 644 1243" data-label="Section-Header"> <h2>La notion de programme revisitée</h2> </div> <div data-bbox="65 1276 526 1494" data-label="List-Group"> <ul style="list-style-type: none"> • En gros, programme = classe principale, contenant un main + classes bibliothèques <ul style="list-style-type: none"> – Utilisateur (ici : Rationnel) – Java (Math, System, etc.) </div>	<div data-bbox="1093 1180 1240 1225" data-label="Section-Header"> <h2>exemple</h2> </div> <div data-bbox="853 1270 1417 1308" data-label="List-Group"> <ul style="list-style-type: none"> • Rationnels et personnes : complet (Ex6.java) </div>
<div data-bbox="218 1756 520 1800" data-label="Section-Header"> <h2>objets et fonctions</h2> </div> <div data-bbox="65 1836 632 2065" data-label="List-Group"> <ul style="list-style-type: none"> • principe • exemple • répartition des fonctions dans les classes • Modification d'objets paramètres de fonctions • allocation dynamique d'objets dans fonctions </div>	<div data-bbox="871 1731 1460 1823" data-label="Section-Header"> <h2>Modification d'objets paramètres de fonctions</h2> </div> <div data-bbox="858 1836 1458 2096" data-label="List-Group"> <ul style="list-style-type: none"> • Comme pour un tableau, une fonction peut <ul style="list-style-type: none"> – Modifier la zone (= les champs) d'un objet passé en paramètre – Créer un objet et le renvoyer (allocation dynamique dans une fonction) • fonctionnement : à nouveau, objet se comporte exactement comme tableau </div>

<div data-bbox="86 31 657 120" data-label="Section-Header"> <h2>Rappel : modifier une variable dans une fonction ?</h2> </div> <div data-bbox="102 125 526 519" data-label="Text"> <pre>class ExFonction2 { static void f(int n) { n = n + 1; } public static void main(String[] args) { int n = 10; f(n); System.out.println(n); // affiche 10 } }</pre> </div>	<div data-bbox="874 31 1465 120" data-label="Section-Header"> <h2>rappel : modifier un tableau passé en argument</h2> </div> <div data-bbox="900 152 1321 510" data-label="Text"> <pre>Static void f(int [] t) { t[0] = t[0] + 1; } public static void main(String[] args) { int [] t = new int[7]; t[0] = 10; f(t); System.out.println(t[0]); // affiche 11 }</pre> </div>
<div data-bbox="79 640 667 685" data-label="Section-Header"> <h2>Modifier un objet passé en argument</h2> </div> <div data-bbox="70 712 676 1075" data-label="List-Group"> <ul style="list-style-type: none"> • Exactement comme tableaux • Une fonction peut modifier les valeurs d'un objet extérieur à elle-même • On dit que les objets sont passés par référence (ou par variable, ou par adresse) • En effet, c'est la référence à l'objet (son adresse) qui est passée à la fonction • Bien sûr, cette adresse est passée par... valeur (recopiée) </div>	<div data-bbox="1104 622 1238 667" data-label="Section-Header"> <h2>exemple</h2> </div> <div data-bbox="865 712 1398 792" data-label="List-Group"> <ul style="list-style-type: none"> • Rationnels : modification d'objet (Ex4.java) <ul style="list-style-type: none"> – déroulement </div>
<div data-bbox="73 1164 137 1196" data-label="Section-Header"> <h2>Cases</h2> </div>	<div data-bbox="1120 1200 1219 1245" data-label="Section-Header"> <h2>rappel</h2> </div> <div data-bbox="868 1281 1445 1469" data-label="List-Group"> <ul style="list-style-type: none"> • Bien sûr, comme on l'a vu, impossible modifier une variable extérieure à une fonction (objet, tableau, int, etc.) • C'est la zone du tableau ou de l'objet qui est modifiée, pas l'adresse </div>
<div data-bbox="226 1760 517 1805" data-label="Section-Header"> <h2>objets et fonctions</h2> </div> <div data-bbox="73 1841 628 2065" data-label="List-Group"> <ul style="list-style-type: none"> • principe • exemple • répartition des fonctions dans les classes • Modification d'objets paramètres de fonctions • allocation dynamique d'objets dans fonctions </div>	<div data-bbox="865 1733 1471 1823" data-label="Section-Header"> <h2>Allocation dynamique d'un objet dans une fonction et son renvoi</h2> </div> <div data-bbox="868 1841 1455 1957" data-label="List-Group"> <ul style="list-style-type: none"> • créer un objet dans une fonction et renvoyer son adresse • Mêmes motivations que pour tableau </div>

<div data-bbox="296 60 445 109" data-label="Section-Header"> <h2>exemple</h2> </div> <div data-bbox="60 154 659 385" data-label="List-Group"> <ul style="list-style-type: none"> • Rationnels : allocation dynamique d'objet : créer un objet à partir de valeurs de la ligne de commande (Ex5.java) <ul style="list-style-type: none"> – Déroulement (avec args) • Un seul appel de fonction dans main (Ex5bis.java) </div>	<div data-bbox="858 44 938 76" data-label="Text"> <p>Cases</p> </div>
<div data-bbox="271 631 467 672" data-label="Section-Header"> <h2>vocabulaire</h2> </div> <div data-bbox="60 714 676 831" data-label="List-Group"> <ul style="list-style-type: none"> • Ce qu'on a appelé « fonction » jusqu'à présent est appelé « méthode de classe » dans la littérature <ul style="list-style-type: none"> – On verra pourquoi bientôt </div>	<div data-bbox="900 638 1436 685" data-label="Section-Header"> <h2>objets et fonctions : récapitulatif</h2> </div> <div data-bbox="858 719 1436 996" data-label="List-Group"> <ul style="list-style-type: none"> • principe • exemple • répartition des fonctions dans les classes • Modification d'objets paramètres de fonctions • allocation dynamique d'objets dans fonctions • (Delannoy 6) </div>
<div data-bbox="279 1198 459 1245" data-label="Section-Header"> <h2>délégués ?</h2> </div>	