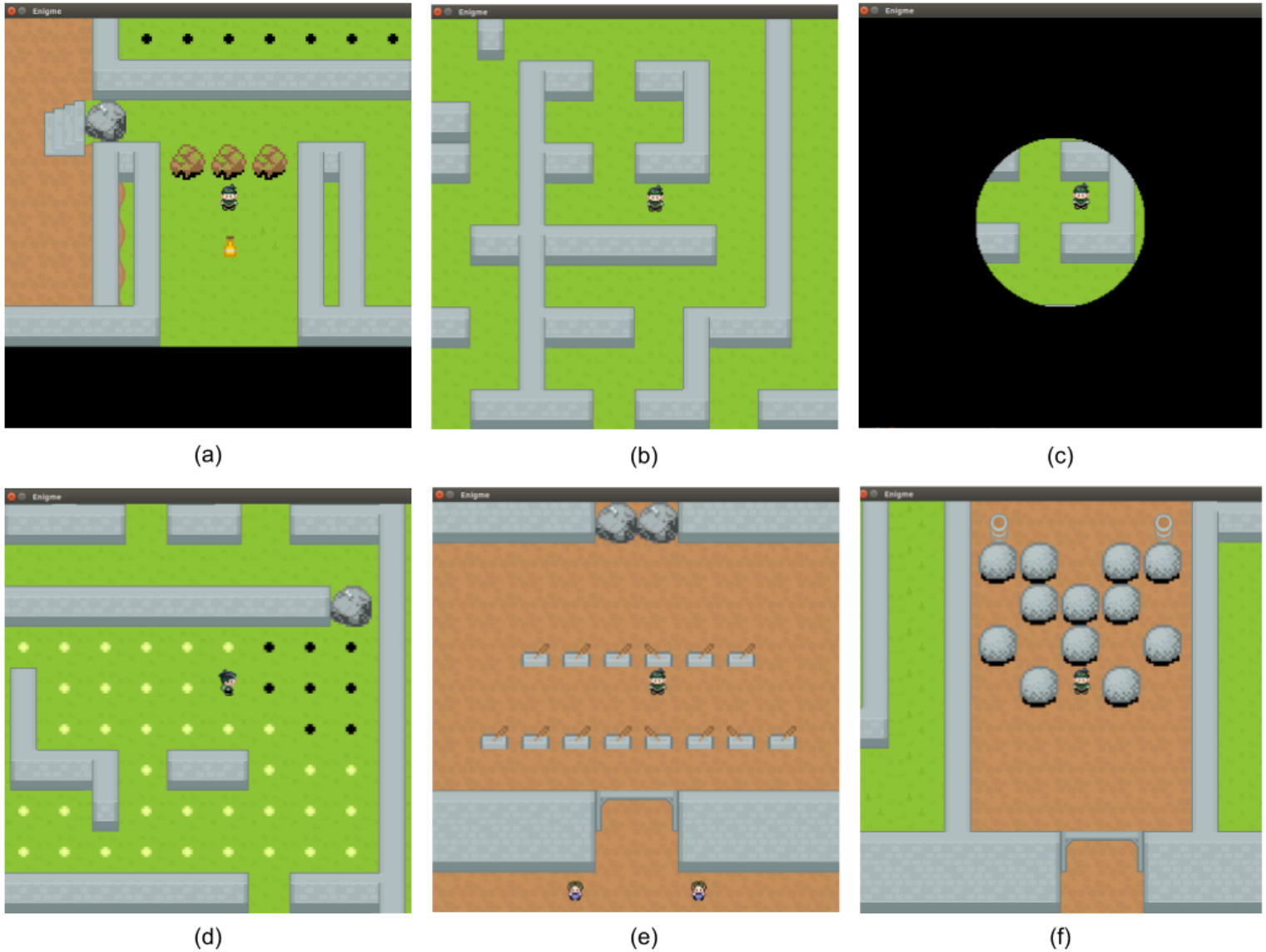
Introduction à la Programmation Objet : Mini-projet 2

Laboratoire d'Intelligence Artificielle
Faculté I&C

J. Sam

«Mini-projet» 2 : les origines



Auteur de la maquette : Bastien Chatelain

«Mini-projet» 2 : au fil du temps

La maquette a permis de réaliser de nombreuses instances de jeux :

- ZeldIC
- IC Super Pacman
- ICWars
- ICRogue
- ICMon
- ICoop
- ...

et cette année ... ICMaze :)

J. Sam

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

«Mini-projet» 2

Objectifs : appréhender les avantages et limitations de l'approche orientée-objet

- en se plaçant à des niveaux d'abstraction suffisamment généraux pour permettre des réutilisation intéressantes
- en intégrant du code existant

J. Sam

Mini-projet 2 : pourquoi des jeux ?

- Terrain d'expérimentation très fertile pour l'orienté objet
- Un nombre infini d'évolutions possibles
- Chaque évolution sous-tend souvent des problématiques de conception intéressantes

☞ Le but est moins le résultat obtenu que ce que l'on apprend sur le chemin pour y parvenir :)

J. Sam

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Consignes de base

Une archive est fournie avec déjà beaucoup de code et des ressources.

- Le code de la maquette (`game-engine`) ne doit pas être modifié
- Ne pas utiliser de paquetages non standards
- Utilisez un dépôt privé si vous utilisez `git`
- L'ajout de ressources est possible, mais le portail limite l'archive à 3MB
  - ☞ Contactez l'équipe du cours pour une dérogation
- Rendu final ouvert
  - ☞ fichiers `README` et `CONCEPTION` à prévoir
- Concours
  - ☞ Petit descriptif à fournir

Consignes méthodologiques

- Regardez la vidéo sur le schéma des interactions avant le TP de vendredi.
- Commencez d'emblée le mini-projet **même si vous n'avez pas fini le tutoriel**.
- Fixez vous des objectifs raisonnables en fonction de votre niveau.
- Lisez l'intégralité des consignes précédant une tâche avant de vous lancer dans le codage.
- Sauvegardez régulièrement.
- Aidez vous du debugger pour comprendre les situations de «crash».
- Si vous travaillez avec git, poussez sur le dépôt toute tâche fonctionnelle avant de passer à la suivante. Soyez très systématique dans l'utilisation de git
  - ☞ Attentions aux usages intempestifs des options de fusion
- Tenez-vous informé.es des «précisions et correctifs».

J. Sam

Consignes méthodologiques – git

- Si vous travaillez avec git, poussez sur le dépôt des états de code **compilables** et si possible **fonctionnels**.
- Poussez de façon **«atomique»**.
- Soyez très systématique : **concertez-vous**, n'oubliez jamais de faire un pull quand vous commencez à travailler et un push quand vous arrêtez.
- Travaillez si possible sur des **parties disjointes** si vous souhaitez travailler en parallèle.
  - ☞ Attention aux usages intempestifs des options de fusion !

J. Sam

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Rendu

Délai : Jeudi 18.12 13 :00

- Protocole de rendu différent du mini-projet 1 (sur la base de jetons)
  - Utilitaire fourni pour constituer une archive `.zip`
  - Soumission depuis un portail disponible sur Moodle
  - Une soumission par groupe

  ☞ le protocole exact de soumission sera donné en temps voulu
- Portail de rendu ouvert 10 jours avant la date limite

  ☞ **Essayez de soumettre dès l'ouverture, même si le projet n'est de loin pas achevé**

Attention ! : vous devez rendre des projets qui compilent. **Seuls les projets compilables seront corrigés**.
Le portail génère un feedback indiquant si le projet compile ou pas.

J. Sam
ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Evaluation

- Évaluation fonctionnelle : les détails de mise en oeuvre sont peu spécifiés et peu importants (documentation dans le README de choix alternatifs)
- Les bugs marginaux ou difficile à reproduire ne sont pas pris en considération
- Évaluation qualitative :
  - Bonne conception OO :
    - Appliquez le principe de «code to an interface not to an implementation».
    - Évitez les failles d'encapsulation.
    - Évitez la prolifération d'attributs protégés.
    - Utilisez le `abstract` à bon escient.
    - Établissez des liens qui font sens entre les classes.
    - Utilisez l'héritage pour éviter des duplications inutiles de code.
  - Critères qualitatifs usuels : lisibilité, modularisation, conventions de nommage, cohérence du style.

J. Sam

Étapes

- Étape 1 : application assez directe du tutoriel
  - ☞ Application du schéma des interaction du tutoriel III
  - ☞ Parer à l'absence du «multiple dispatch»
  - ☞ Découverte d'un premier «patron de conception»

  Plus complexe mais très guidé

- Étape 2 : Génération de labyrinthes
  - ☞ Récursion

Les étapes sont jalonnées de tâches intermédiaires pour **avancer de façon robuste petit pas par petit pas**.

Le code des graphismes et animations est fourni.

J. Sam

Étapes (suite)

- Étape 3 : Adversaires et batailles
  - Concept d'«automate à états finis» (logique des combats)
  - Générations procédurales de niveaux (augmentation de la difficulté des labyrinthes par paliers)
- Étape 4 : le 1/2 point pour arriver au 6
  - ☞ Beaucoup moins guidée et va vous «challenger» l'encapsulation
- Étape 5 : 20 points d'extensions à choix
  - ☞ libre et très peu guidé

Les ajouts sont assez modulaires (possibilité de travailler en parallèle sur certains aspects du projets)

Nature de l'aide et forum Ed

- Les assistant.es n'ont pas pour tâche de résoudre vos erreurs d'exécution ;
- **Il ne faut pas les contacter en privé** (par exemple sur Telegram) : le seul canal de discussion doit être le forum Ed.
- Ils/elles peuvent vous aider sur les fautes de compilation ;
- et vous aider à développer des stratégies pour débusquer les sources des erreurs d'exécution par vous même ;
- il y a une équipe limitée en appui : **privilégiez le forum Ed pendant la semaine !**

J. Sam

Bonnes pratiques sur Ed

- **Ne pas publier votre code** (utilisez un post privé s'il est nécessaire de montrer votre code et ne le faites qu'à notre demande).
- Avant de montrer du code, essayer de circonscrire les conditions d'occurence du bug, en nous indiquant par exemple, ce que vous observez dans le debugger.
- Donnez des **titres parlants** à vos posts.
- **Cherchez si la question n'a pas déjà été posée**.

J. Sam

«Mini-projet» 2

Bon codage !

J. Sam

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Récursion

La plupart des problèmes (par exemple, le tri d'un tableau, le calcul d'une équation), peuvent être résolus de différentes façons :

☞ différents **algorithmes**

Une catégorie **particulière** de solutions est constituée des méthodes de résolution **récursives**.

Le principe de l'approche récursive est de

> *ramener le problème à résoudre à un sous-problème, version simplifiée du problème d'origine.*

☞ Exemple en mathématiques : le raisonnement par récurrence

J. Sam

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Exemple simple

Calculer la somme des n premiers entiers.

Si je sais le faire pour n, je sais le faire pour $n+1$:

$$S(n+1) = (n+1) + S(n)$$

Condition d'arrêt :

Je sais le faire pour $n = 0$: $S(0) = 0$

Algorithme :

somme

entrée : n
sortie : $S(n)$

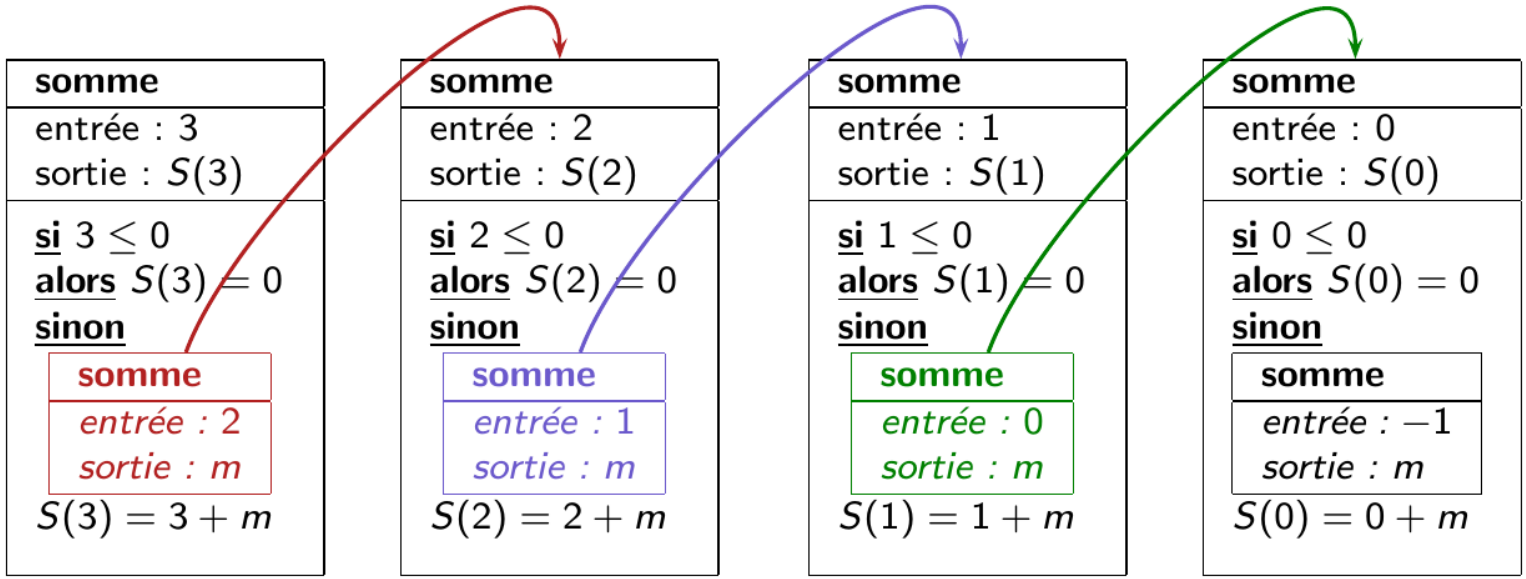
si $n \leq 0$
alors $S(n) \leftarrow 0$
sinon

> ***somme***
>
> *entrée : $n-1$*
> *sortie : m*

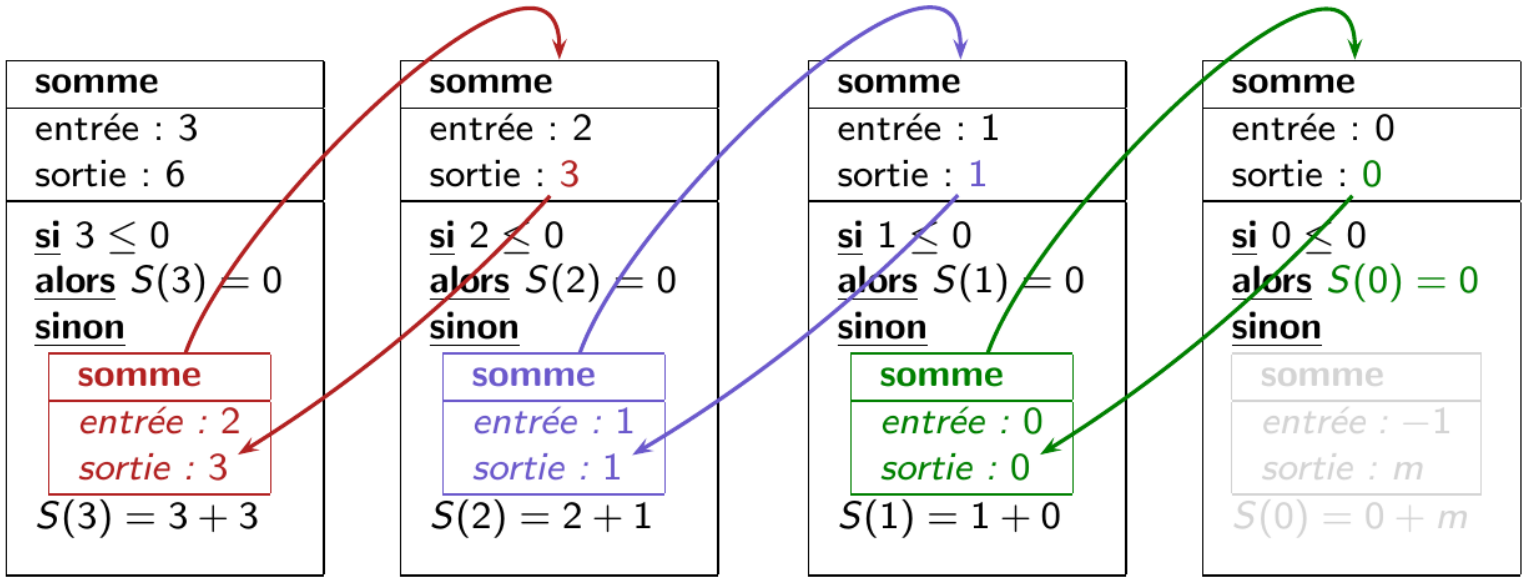
$S(n) \leftarrow n + m$

J. Sam

Somme récursive : empilement des appels



Somme récursive : dépilement des appels



Somme récursive en Java

```
int somme(int n) {
  if (n <= 0) // condition arret
    return 0;
  else
    return (n + somme(n-1));
}
```

Les fonctions récursives

Le schéma général d'une fonction récursive est donc le suivant :

```
type nom(type1 arg1, type2 arg2, ... ) {
  if (terminaison(arg1, arg2, ...)) {
    ...
  } else {
    type1 z; // si nécessaire pour
    type2 y1; //    des calculs intermédiaires
    type2 y2;
    ...
    z = nom(y1, y2, ...)
    ...
  }
}
```

même nom

Pour conclure

La solution récursive n'est pas toujours la seule solution et n'est pas toujours la plus efficace...

...mais elle est parfois beaucoup **plus simple** et/ou **plus pratique** à mettre en œuvre !

Exemples : tris, traitement de structures de données récursives (e.g., arbres, graphes, . . .), . . .

J. Sam

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE