IFT1015 Programmation 1

Traitement de texte

Marc Feeley

Université de Montréal

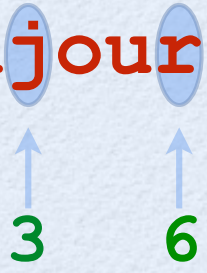
Méthodes importantes sur les textes

- *t*`.length` *(méthode spéciale sans paramètres)*
  - retourne le nombre de caractères dans le texte *t*
- *t*`.charAt(`*i*`)`
  - retourne un texte de longueur 1, le caractère à l'index *i* du texte *t* (peut aussi faire *t*`[`*i*`]`)
- *t*`.charCodeAt(`*i*`)`
  - retourne un entier qui est le **code Unicode** du caractère à l'index *i* du texte *t*
- *t*`.slice(`*i*`,`*j*`)`
  - retourne un texte qui contient les caractères de *t* entre l'index *i* (inclus) et *j* (exclus)

Extraction de sous-textes

- Soit *t* un texte
- *t*`.charAt(`*i*`)` retourne le caractère à la position *i*
  (les indices de caractères vont de 0 à longueur-1)
  Exemple: `"Bonjour".charAt(3)` vaut `"j"`
  `"Bonjour".charAt(6)` vaut `"r"`
  `"Bonjour".charCodeAt(0)` vaut `66`
- *t*`.slice(`*debut*`,` *fin*`)` retourne un sous-texte

  Exemple : `"Bonjour".slice(3,6)` vaut `"jou"`



Similarité des méthodes sur les textes et tableaux

```
> var t = "ABCD";

> t.length
  4

> t.charAt(1)    // ou t[1]
  "B"

> t.slice(1,3);
  "BC"

> t+"EF";
  "ABCDEF"

> t.charCodeAt(0);
  65
```

```
> var t = ["A","B","C","D"];

> t.length
  4

> t[1]
  "B"

> t.slice(1,3);
  ["B", "C"]

> t.concat(["E","F"]);
  ["A", "B", "C", "D", "E", "F"]

> t.concat(["EF"]);
  ["A", "B", "C", "D", "EF"]
```

Exemple : encodagePositionnel

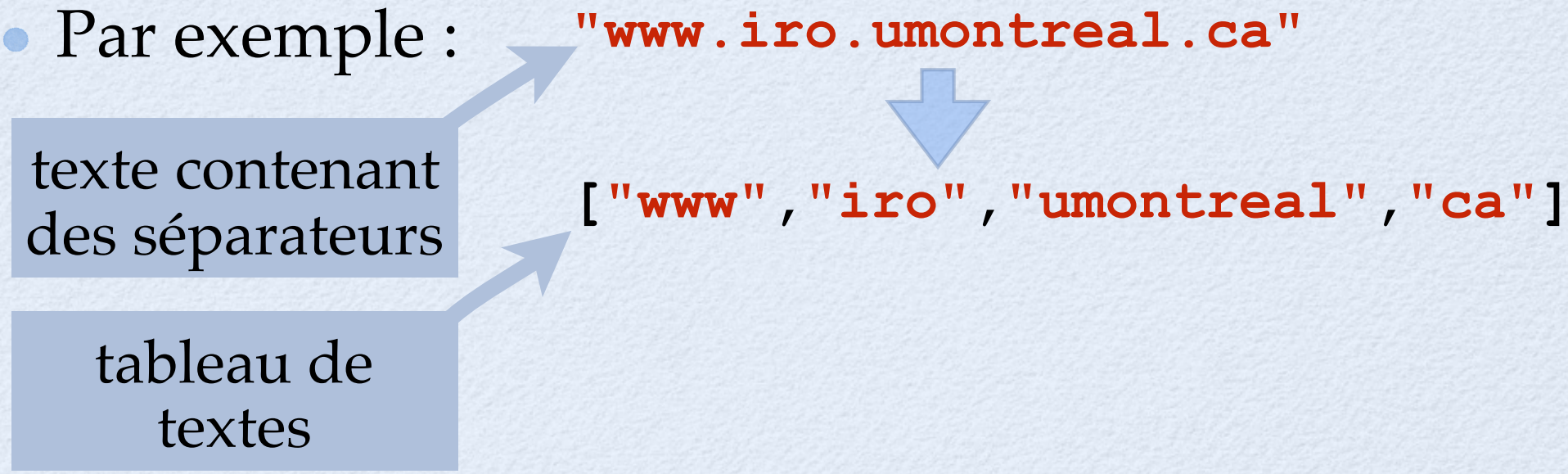
```
var encodagePositionnel = function (n, base) {

    // Cette fonction retourne l'encodage du nombre n dans la
    // base spécifiée, qui peut être entre 2 et 36.

    var chiffres = "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ";

    var e = "";                          // pour accumuler l'encodage

    do {
        e = chiffres.charAt(n % base) + e; // accumuler un chiffre
        n = Math.floor(n / base); // passer aux autres chiffres
    } while (n > 0);                     // tant qu'il reste des chiffres

    return e;
};

print( "0x" + encodagePositionnel(51966, 16) );  // imprime 0xCAFE
```

Exemple : encodagePositionnel

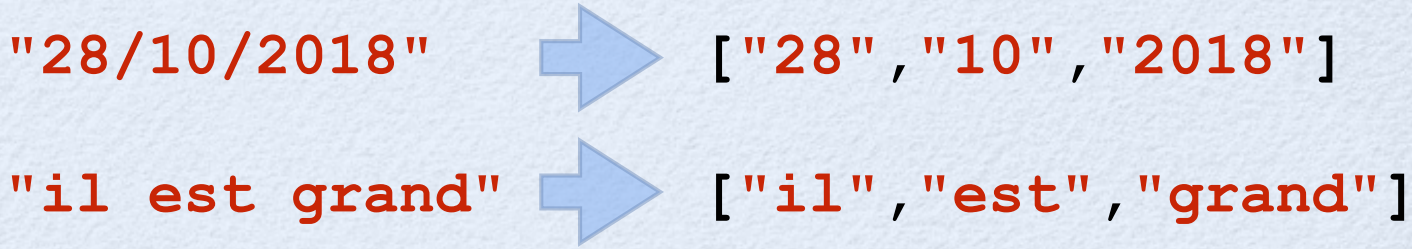
```
var encodagePositionnel = function (n, base) {

    // Cette fonction retourne l'encodage du nombre n dans la
    // base spécifiée, qui peut être entre 2 et 36.

    var chiffres = "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ";

    var e = "";                          // pour accumuler l'encodage

    do {
        var c = n % base;                // le prochain chiffre
        e = chiffres.slice(c, c+1) + e; // accumuler un chiffre
        n = Math.floor(n / base); // passer aux autres chiffres
    } while (n > 0);                     // tant qu'il reste des chiffres

    return e;
};

print( "0x" + encodagePositionnel(51966, 16) );  // imprime 0xCAFE
```

Exemple : découper un texte

- Il est souvent nécessaire de découper un texte en parties (mots, lignes, etc)
- Par exemple :



autres exemples



Exemple : découper un texte

```
var decouper = function (texte, separateur) {
    var resultat = [];
    var debut = 0;
    var i = 0;
    while (i<texte.length) {
        if (texte.charAt(i) == separateur) {
            resultat.push(texte.slice(debut,i));
            debut = i+1;
        }
        i++;
    }
    resultat.push(texte.slice(debut,i));
    return resultat;
};

print( decouper("www.iro.umontreal.ca", ".") );
```

Exemple : découper en mots

- Un “mot” est une séquence de lettres
- Par exemple :

texte contenant des mots

```
"oui... j'ai sept chats!"
```

```
["oui","j","ai","sept","chats"]
```

tableau des mots

Exemple : découper en mots

```
var lettre = function (car) {
    return (car >= "a" && car <= "z") ||
           (car >= "A" && car <= "Z");
};

var decouperEnMots = function (texte) {
    var resultat = [];
    var debut = 0;
    while (debut < texte.length) {
       if (lettre(texte.charAt(debut))) {
            var i = debut+1;
            while (i < texte.length && lettre(texte.charAt(i))) {
                i++;
            }
            resultat.push(texte.slice(debut, i));
            debut = i+1;
        } else {
            debut++;
        }
    }
    return resultat;
};

print( decouperEnMots("oui... j'ai sept chats!") );
```

Exemple : chercher un mot

- Trouver la position où se trouve un mot
- Par exemple :

```
positionMot("Je connais Java et JavaScript", "Java") => 11
```

Exemple : chercher un mot

version avec **charAt**

```
var positionMot = function (texte, mot) {
    prochainePos:
    for (var i=0; i<=texte.length-mot.length; i++) {
        for (var j=0; j<mot.length; j++) {
            if (texte.charAt(i+j) != mot.charAt(j)) {
                continue prochainePos;
            }
        }
        return i; // mot trouvé!
    }
    return -1; // indiquer échec
};

print( positionMot("Je connais Java et JavaScript", "Java") );
```

Exemple : chercher un mot

version avec **slice**

```
var positionMot = function (texte, mot) {
    for (var i=0; i<=texte.length-mot.length; i++) {
        if (texte.slice(i, i+mot.length) == mot) {
            return i; // mot trouvé!
        }
    }
    return -1; // indiquer échec
};

print( positionMot("Je connais Java et JavaScript", "Java") );
```

La méthode indexOf

- Pour rechercher la position d'une valeur dans un tableau on peut se servir des méthodes **`indexOf`** et **`lastIndexOf`**
- *tableau* **`.indexOf(`***valeur***`)`**
  - retourne l'index (le plus **bas**) où se trouve *valeur* dans *tableau* (-1 si la valeur n'existe pas dans *tableau*)
- *tableau* **`.lastIndexOf(`***valeur***`)`**
  - retourne l'index (le plus **haut**) où se trouve *valeur* dans *tableau* (-1 si la valeur n'existe pas dans *tableau*)

La méthode indexOf

- La méthode **`indexOf`** fait donc le même travail que la fonction **`position`** :

```
var position = function (t, val) {

    for (var i=0; i<t.length; i++) {
        if (t[i] == val) {
            return i;  // on a trouvé l'index!
        }
    }

    return -1; // code indiquant échec
};
```

tableau`.indexOf(`*valeur*`)` ≣ `position(`*tableau*`,`*valeur*`)`

La méthode lastIndexOf

- La méthode **`lastIndexOf`** fait une recherche descendante comme la fonction **`lastPosition`** :

```
var lastPosition = function (t, val) {

    for (var i=t.length-1; i>=0; i--) {
        if (t[i] == val) {
            return i;  // on a trouvé l'index!
        }
    }

    return -1; // code indiquant échec
};
```

tableau`.lastIndexOf(`*valeur*`)` ≣ `lastPosition(`*tableau*`,`*valeur*`)`

La méthode indexOf

- On peut indiquer le départ de la recherche aux méthodes **`indexOf`** et **`lastIndexOf`**
- *tableau* **`.indexOf(`***valeur* **`,`***départ***`)`**
  - retourne l'index le plus **bas** et >= *départ* où se trouve *valeur* dans *tableau* (-1 si la valeur n'est pas trouvée)
- *tableau* **`.lastIndexOf(`***valeur* **`,`***départ***`)`**
  - retourne l'index le plus **haut** et <= *départ* où se trouve *valeur* dans *tableau* (-1 si la valeur n'est pas trouvée)

La méthode indexOf

- Les méthodes **`indexOf`** et **`lastIndexOf`** existent aussi sur les textes :
- *texte* **`.indexOf(`***texte2***`)`**
  - retourne l'index (le plus **bas**) où se trouve *texte2* dans *texte* (-1 si la valeur n'existe pas dans *texte*)
- *texte* **`.lastIndexOf(`***texte2***`)`**
  - retourne l'index (le plus **haut**) où se trouve *texte2* dans *texte* (-1 si la valeur n'existe pas dans *texte*)

La méthode indexOf

- La méthode **`indexOf`** sur les textes fait donc le même travail que la fonction **`positionMot`** :

```
var positionMot = function (texte, texte2) {

    for (var i=0; i<=texte.length-texte2.length; i++) {
        if (texte.slice(i, i+texte2.length) == texte2) {
            return i;  // on a trouvé l'index!
        }
    }

    return -1; // code indiquant échec
};
```

texte`.indexOf(`*texte2*`)` ≣ `positionMot(`*texte*`,`*texte2*`)`

La méthode indexOf

- On peut indiquer le départ de la recherche aux méthodes **`indexOf`** et **`lastIndexOf`** sur les textes

`"abracadabra".indexOf("bra",2)` vaut **`8`**

Exemple : positions d'un mot

- Trouver **les positions** où se trouve un mot
- Utiliser un tableau de positions, par exemple :

```
positionsMot("Je connais Java et JavaScript", "Java")
```

```
[11,19]
```

Exemple : positions d'un mot

version avec **slice**

```
var positionsMot = function (texte, mot) {
    var resultat = [];
    for (var i=0; i<=texte.length-mot.length; i++) {
        if (texte.slice(i, i+mot.length) == mot) {
            resultat.push(i); // mot trouvé!
        }
    }
    return resultat;
};

print( positionsMot("Je connais Java et JavaScript", "Java") );
```

Exemple : positions d'un mot

version 2 avec **slice**

```
var positionsMot = function (texte, mot) {
    var resultat = [];
    for (var i=0; i<=texte.length-mot.length; i++) {
        if (texte.slice(i, i+mot.length) == mot) {
            resultat.push(i); // mot trouvé!
            i += mot.length-1;
        }
    }
    return resultat;
};

print( positionsMot("La CAA et AAA sont partenaires", "AA") );
```

évite de trouver un 3e AA ici

Exemple : positions d'un mot

version avec **indexOf**

```
var positionsMot = function (texte, mot) {
    var resultat = [];
    var i = 0;
    while (true) {
        i = texte.indexOf(mot, i);
        if (i == -1) break;
        resultat.push(i); // mot trouvé!
        i += mot.length;
    }
    return resultat;
};

print( positionsMot("La CAA et AAA sont partenaires", "AA") );
```

point de départ de la prochaine recherche du mot

Exemple : remplacer un mot

- Dans un texte, remplacer un mot par un autre, par exemple :

```
remplacerMot("il a deux chats et deux chiens","deux","dix")
```

```
"il a dix chats et dix chiens"
```

Exemple : remplacer un mot

version avec **positionsMot**

```
var remplacerMot = function (texte, mot, autre) {
    var positions = positionsMot(texte, mot);
    var resultat = "";
    var debut = 0;
    for (var i=0; i<positions.length; i++) {
        resultat += texte.slice(debut, positions[i]) + autre;
        debut = positions[i] + mot.length;
    }
    resultat += texte.slice(debut, texte.length);
    return resultat;
};

print( remplacerMot("il a deux chats et deux chiens","deux","dix") );
```

Les méthodes split et join

- Les méthodes **`split`** et **`join`** permettent de découper et former un texte avec un séparateur
- *texte* **`.split(`***texte2***`)`**
  - retourne un tableau contenant les textes issus d'un découpage de *texte* au séparateur *texte2*

    `"abXYcdXYef".split("XY")` vaut **`["ab","cd","ef"]`**
- *tableau* **`.join(`***texte***`)`**
  - retourne un texte formé de la concaténation des textes dans *tableau* séparés par *texte*

    `["ab","cd","ef"].join("XY")` vaut **`"abXYcdXYef"`**

Exemple : remplacer un mot

version avec **`split`** et **`join`**

```
var remplacerMot = function (texte, mot, autre) {
    return texte.split(mot).join(autre);
};

print( remplacerMot("j'ai deux chats et deux chiens","deux","dix") );
```

Énoncé switch

L'énoncé switch

- L' **énoncé `switch`** permet de faire une **exécution conditionnelle** d'un de plusieurs énoncés en fonction de la valeur d'une expression
- Syntaxe de base :

```
switch (‹expression›0) {
    case ‹expression›1: ‹énoncés›1
    case ‹expression›2: ‹énoncés›2
    case ‹expression›3: ‹énoncés›3
    …
}
```

corps du **`switch`**

- L'évaluation de ‹*expression*›$_0$ se fait en premier
- Le premier groupe d'énoncés qui a une expression de même valeur est exécuté (puis ceux qui suivent)

Exemples

```
// Fichier: choix1.js

// Ce programme illustre l'énoncé switch.

var choix = function (x) {

    switch (x/10) {
        case 1: print("a");
        case 2: print("b");
        case 3: print("c");
        case 4: print("d");
        case 5: print("e");
   }
};

choix(50); // imprime: e

choix(30); // imprime: c d e

choix(0);  // imprime rien
choix(90); // imprime rien
```

Exemples

```
// Fichier: choix2.js

// Ce programme illustre les énoncés switch et break.

var choix = function (x) {

    switch (x/10) {
        case 1: print("a"); break;
        case 2: print("b"); break;
        case 3: print("c"); break;
        case 4: print("d"); break;
        case 5: print("e"); break;
   }
};

choix(50); // imprime: e

choix(30); // imprime: c

choix(0);  // imprime rien
choix(90); // imprime rien
```

Un **break** exécuté dans un **switch** sort immédiatement du **switch**

Exemples

```
// Fichier: choix3.js

// Ce programme illustre le switch avec default.

var choix = function (x) {

    switch (x/10) {
        case 1:  print("a"); break;
        case 2:  print("b"); break;
        case 3:  print("c"); break;
        case 4:  print("d"); break;
        default: print("e"); break;
   }
};

choix(50); // imprime: e

choix(30); // imprime: c

choix(0);  // imprime: e
choix(90); // imprime: e
```

Le cas **default** est exécuté lorsqu'aucun autre cas concorde

Exemples

```
// Fichier: choix3-avec-if.js

// Ce programme est équivalent à choix3.js .

var choix = function (x) {
    var n = x/10;
    if (n === 1) {
        print("a");
    } else if (n === 2) {
        print("b");
    } else if (n === 3) {
        print("c");
    } else if (n === 4) {
        print("d");
    } else {
        print("e");
    }
};

choix(50); // imprime: e
choix(30); // imprime: c
choix(0);  // imprime: e
choix(90); // imprime: e
```

Un **switch** peut être remplacé par une cascade de **if**s

Exemples

```
// Fichier: choix4.js

// Ce programme illustre le switch avec des cas texte.

var traduire = function (x) {

    switch (x) {
        case "un":      return 1;
        case "deux":    return 2;
        case "trois":   return 3;
        case "quatre": return 4;
        default:        return -1;
    }
};

print( traduire("trois") ); // imprime: 3

print( traduire("dix") );   // imprime: -1
```

Les cas sont des expressions de n'importe quel type

Exemples

```
// Fichier: choix5.js

// Ce programme illustre le switch avec des cas texte.

var traduire = function (x) {

    switch (x) {
        case "un":      return "one";
        case "deux":    return "two";
        case "trois":   return "three";
        case "quatre": return "four";
        default:        return "other";
   }
};

print( traduire("trois") ); // imprime: three

print( traduire("dix") );   // imprime: other
```

traduire du français à l'anglais

Exemples

```
// Fichier: choix6.js

// Ce programme illustre le switch avec des cas texte.

var traduire = function (x) {

    switch (x) {
        case "uno":
        case "un":      return "one";
        case "dos":
        case "deux":    return "two";
        case "tres":
        case "trois":   return "three";
        case "cuatro":
        case "quatre": return "four";
        default:        return "other";
    }
};

print( traduire("trois") ); // imprime: three

print( traduire("dos") );   // imprime: two
```

traduire du français et espagnol à l'anglais

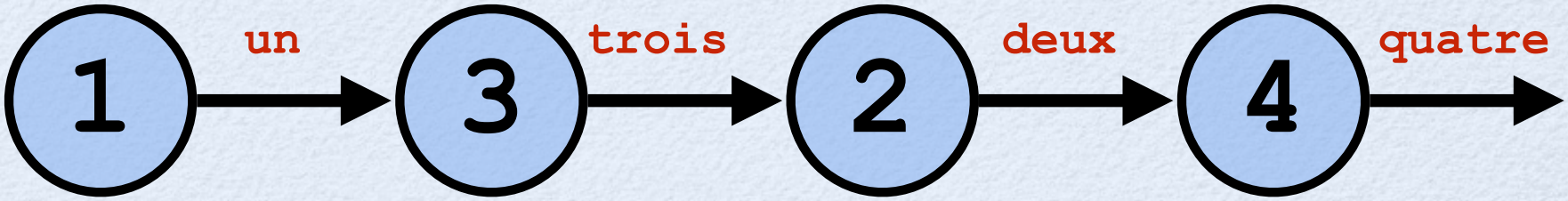
Automates

```
// Fichier: automate1.js

// Utilisation de switch pour réaliser un automate.

var automate = function (etat) {

    while (true) {
        switch (etat) {
            case 1: print("un");      etat = 3; break;
            case 2: print("deux");    etat = 4; break;
            case 3: print("trois");   etat = 2; break;
            case 4: print("quatre"); return;
        }
    }
};

automate(1); // imprime?
```

imprime :
un trois deux quatre

Automates

```
// Fichier: automate1.js

// Utilisation de switch pour réaliser un automate.

var automate = function (etat) {

    while (true) {
        switch (etat) {
            case 1: print("un");      etat = 3; break;
            case 2: print("deux");    etat = 4; break;
            case 3: print("trois");   etat = 2; break;
            case 4: print("quatre"); return;
        }
    }
};

automate(1); // imprime?
```



Exemple : valider un texte

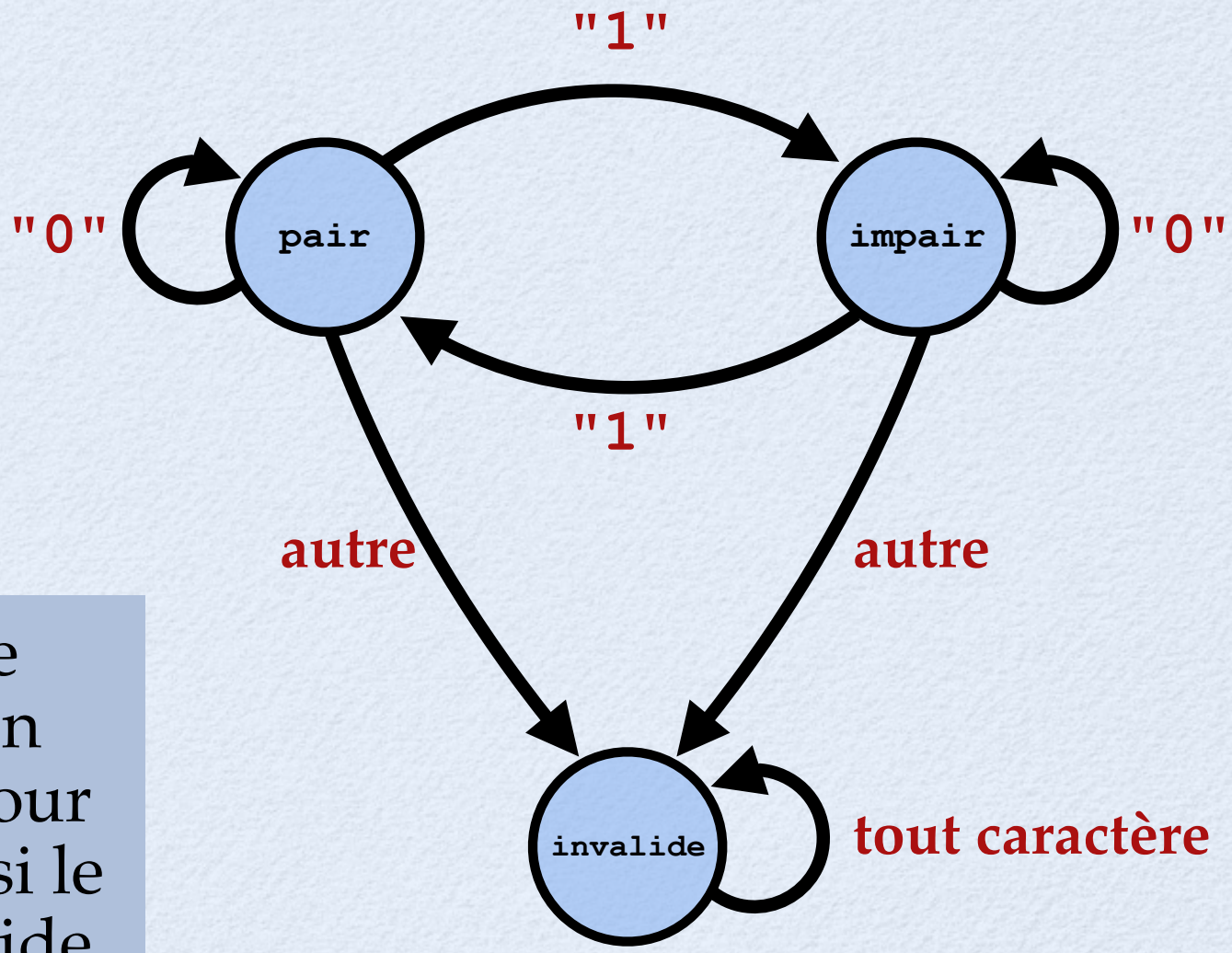
- Specification : on désire une fonction **`valide`** qui prend un texte en paramètre et qui retourne un booléen indiquant si le texte est conforme au format suivant : ***il contient aucun autre caractère que 0 et 1 et un nombre pair de 1***
- Tests unitaires :

```
var testValide = function () {  // tests unitaires
    assert( valide("10010") == true  );
    assert( valide("1 1")   == false );
    assert( valide("1101")  == false );
    assert( valide("1")     == false );
    assert( valide("11")    == true  );
    assert( valide("")      == true  );
};
```

Exemple : valider un texte

```
// Fichier: valide1.js

// Validation d'un texte binaire à parité paire.

var valide = function (t) {

    var n = 0;

    for (var i=0; i<t.length; i++) {
        switch (t.charAt(i)) {
            case "0": break;
            case "1": n++; break;
            default:  return false;
        }
    }

    return (n % 2) == 0;
};
```

Exemple : valider un texte



Approche utilisant un automate pour déterminer si le texte est valide

Exemple : valider un texte

```
var valide = function (t) {

    var etat = "pair";

    for (var i=0; i<t.length; i++) {
        switch (etat) {
            case "pair":
                switch (t.charAt(i)) {
                    case "0": continue;
                    case "1": etat = "impair"; continue;
                    default:  etat = "invalide"; continue;
                }
            case "impair":
                switch (t.charAt(i)) {
                    case "0": continue;
                    case "1": etat = "pair"; continue;
                    default:  etat = "invalide"; continue;
                }
            case "invalide":
                continue;
        }
    }

    return etat == "pair";
};
```

Exemple : valider un texte

```
// Fichier: valide2.js

// Validation d'un texte binaire à parité paire.

var valide = function (t) {

    var pair = true;

    for (var i=0; i<t.length; i++) {
        switch (t.charAt(i)) {
            case "0": break;
            case "1": switch (pair) {
                          case true:  pair = false; break;
                          case false: pair = true;  break;
                      }
                      break;
            default:  return false;
        }
    }

    return pair;
};
```

Exemple : valider un texte

```
// Fichier: valide3.js

// Validation d'un texte binaire à parité paire.

var valide = function (t) {

    var pair = true;

    for (var i=0; i<t.length; i++) {
        switch (t.charAt(i)) {
            case "0": break;
            case "1": pair = !pair; break;
            default:  return false;
        }
    }

    return pair;
};
```

Exemple : nombres romains

- Specification : on désire une fonction **`romain`** qui prend un texte en paramètre et qui retourne un entier. Cet entier est -1 si le texte n'est pas un nombre romain entre 1 et 3999, sinon l'entier retourné est la valeur du nombre romain.
- Tests unitaires :

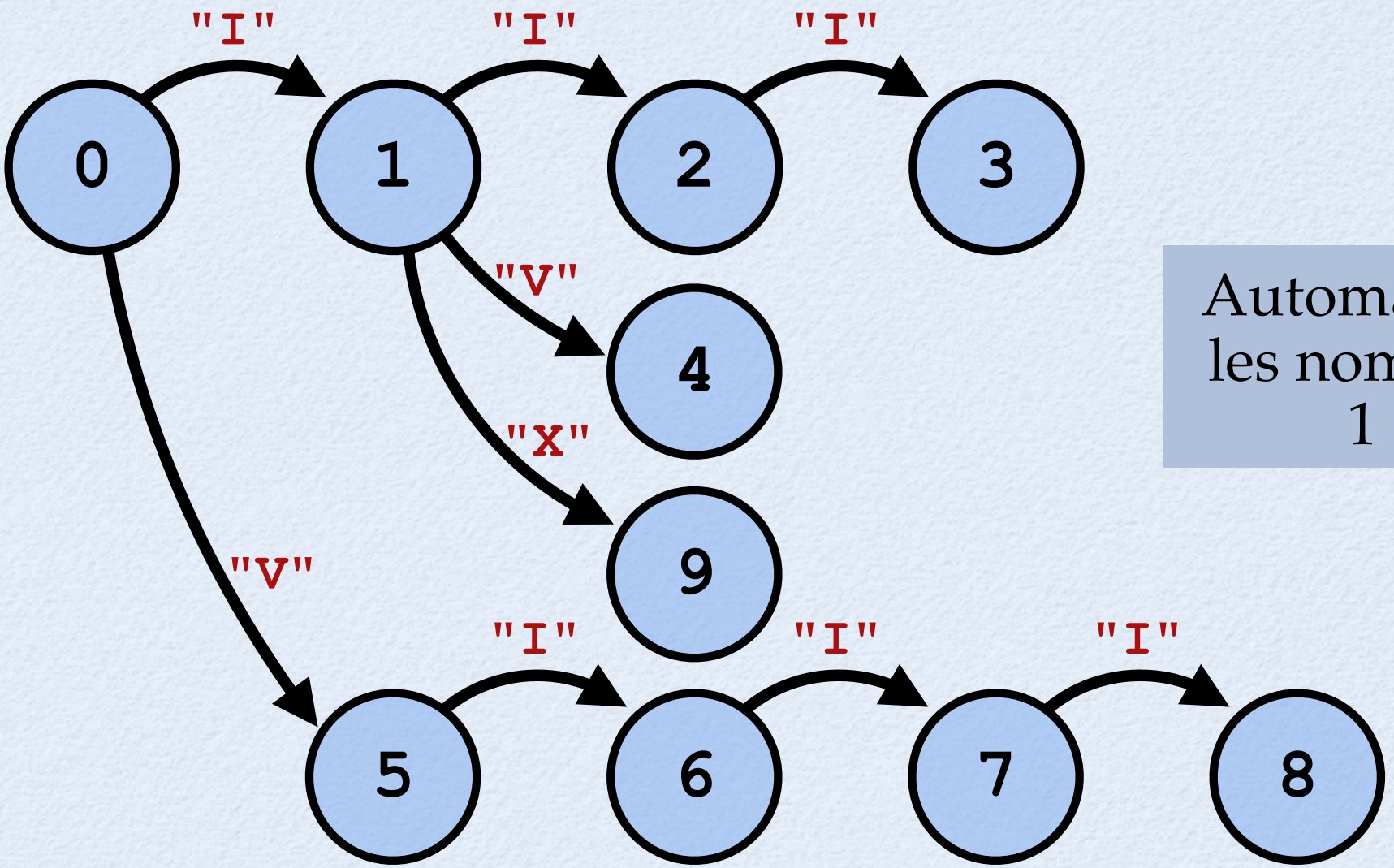
```
var testRomain = function () {  // tests unitaires
    assert( romain("I")           == 1    );
    assert( romain("XX")          == 20   );
    assert( romain("CIV")         == 104  );
    assert( romain("CCCLXXX")     == 380  );
    assert( romain("MCMVII")      == 1907 );
    assert( romain("MMMCMXCIX") == 3999 );
    assert( romain("CIL")         == -1   );
    assert( romain("Z")           == -1   );
    assert( romain("")            == -1   );
};
```

Exemple : nombres romains

- Rappel :

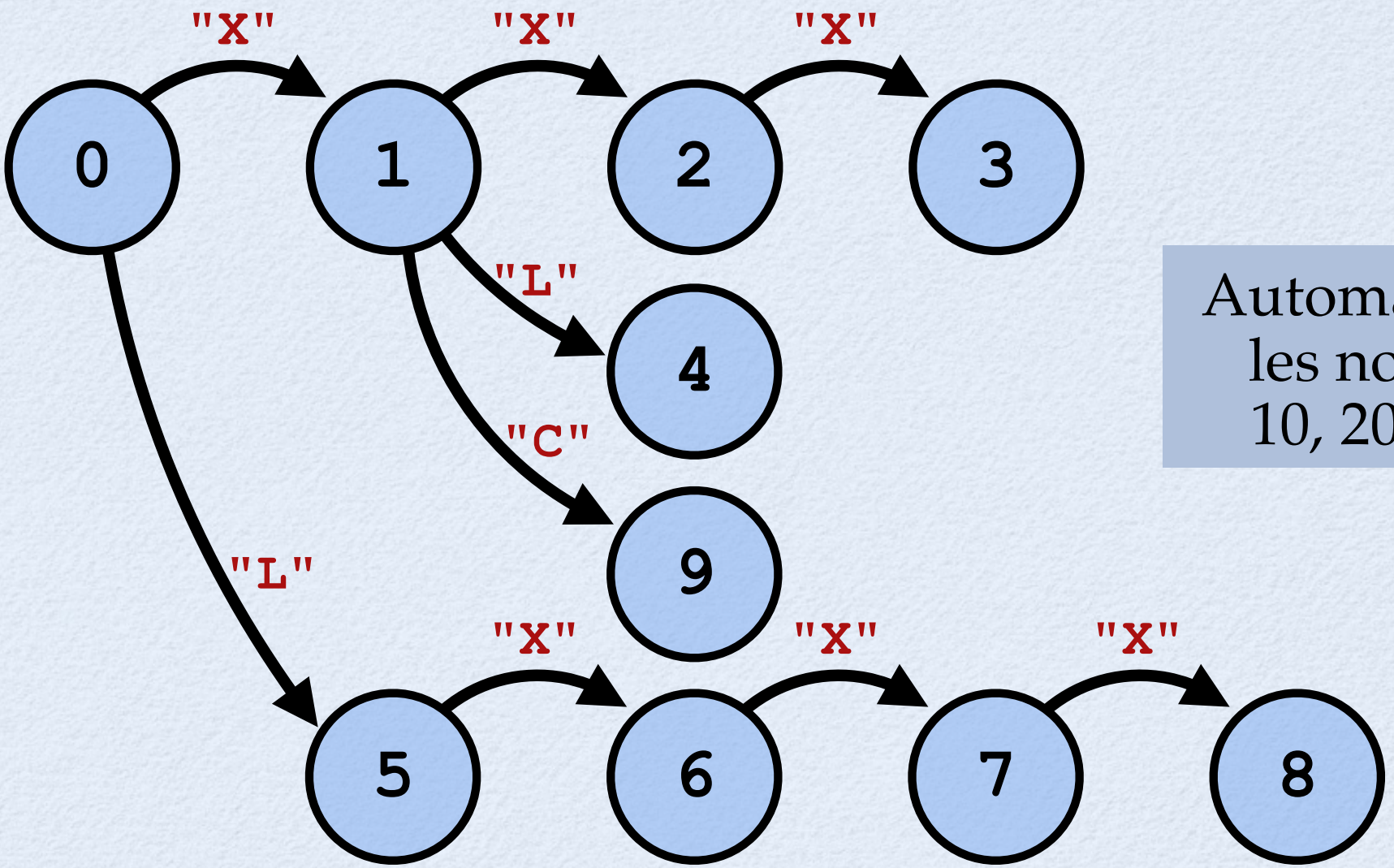
1000	M	100	C	10	X	1	I
2000	MM	200	CC	20	XX	2	II
3000	MMM	300	CCC	30	XXX	3	III
		400	CD	40	XL	4	IV
		500	D	50	L	5	V
		600	DC	60	LX	6	VI
		700	DCC	70	LXX	7	VII
		800	DCCC	80	LXXX	8	VIII
		900	CM	90	XC	9	IX

Exemple : nombres romains



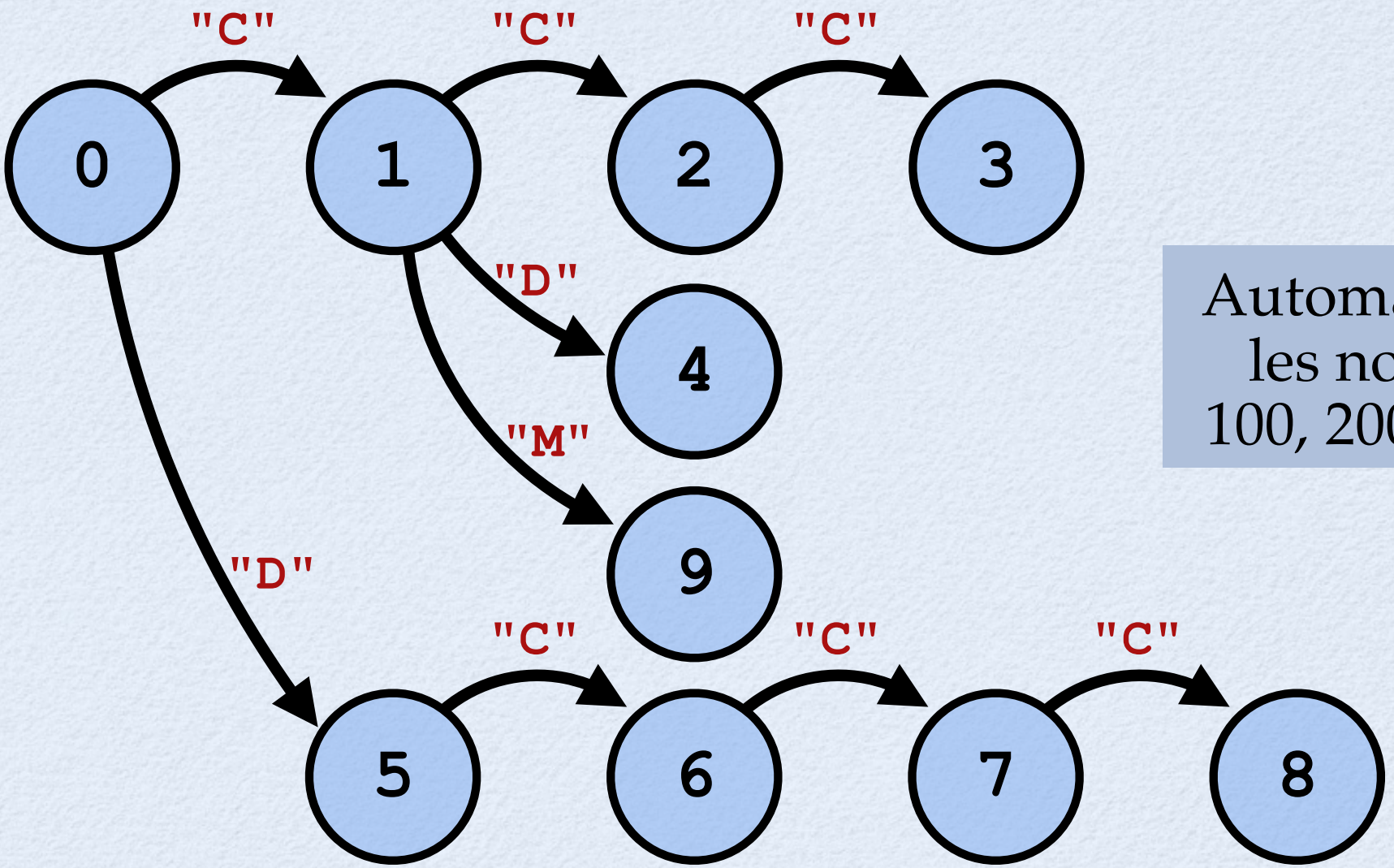
Automate pour les nombres de 1 à 9

Exemple : nombres romains



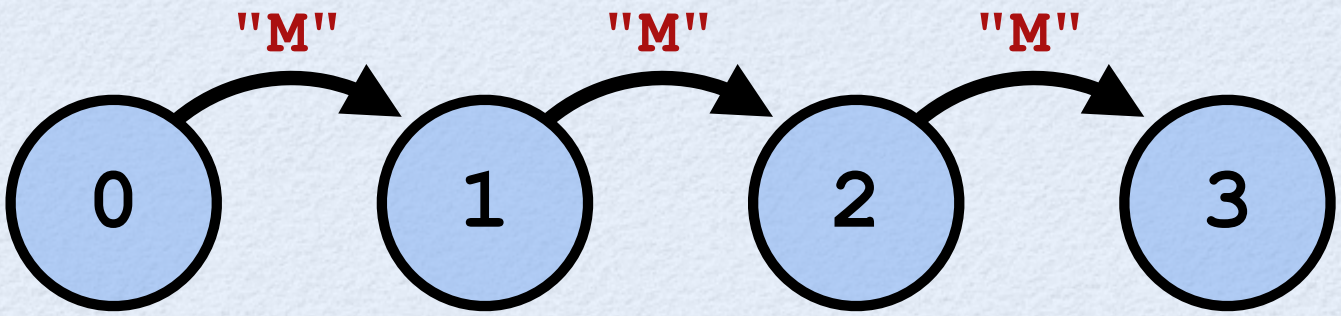
Automate pour les nombres 10, 20, … 90

Exemple : nombres romains



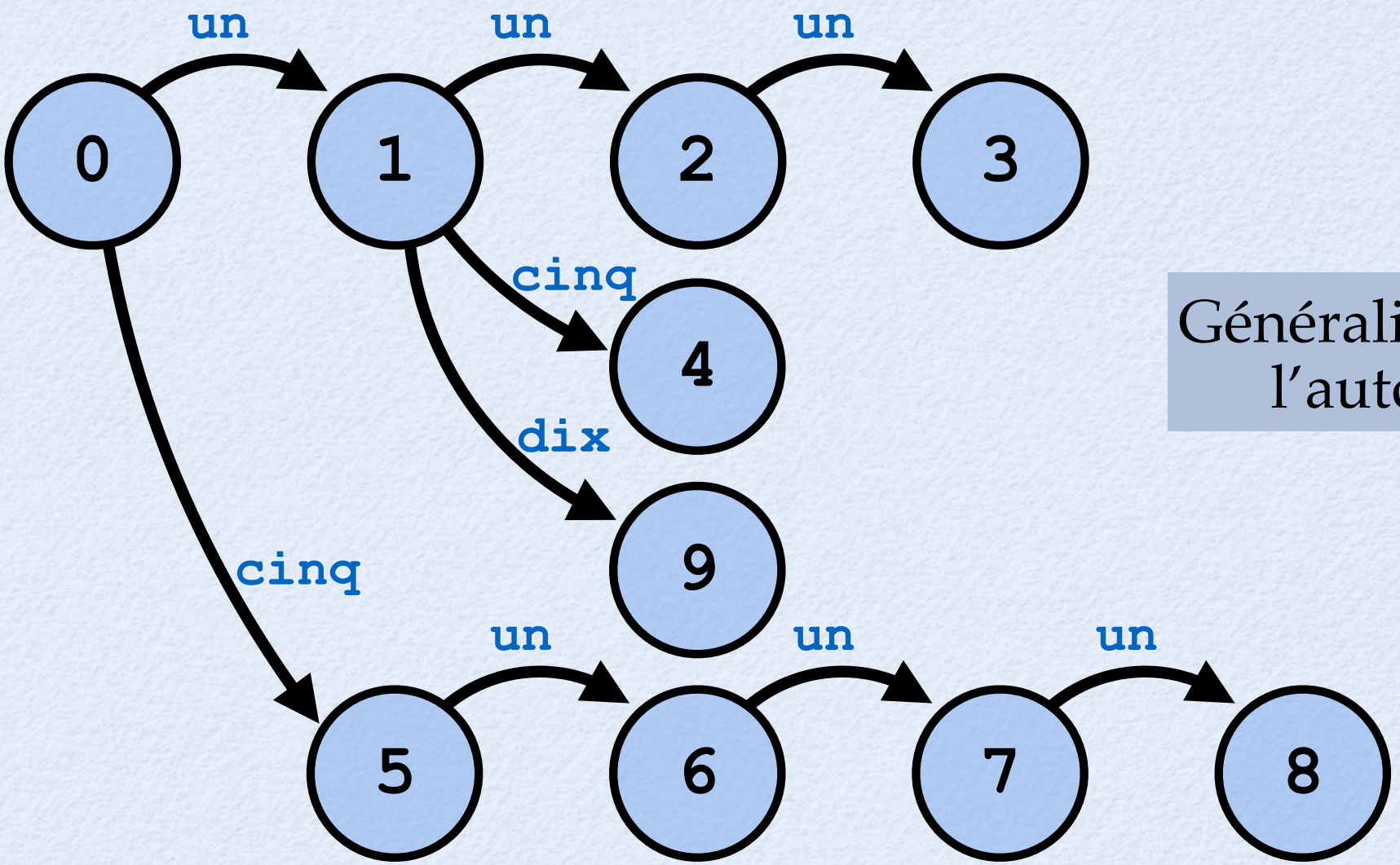
Automate pour les nombres 100, 200, … 900

Exemple : nombres romains



Automate pour les nombres 1000, 2000, 3000

Exemple : nombres romains



Généralisation de l'automate

```
var romain = function (t) {

    var pos = 0;

    var prochain = function () {
        if (pos < t.length) {
            return t.charAt(pos);
        } else {
            return " ";
        }
    };

    var chiffre = function (un, cinq, dix) {

        var val = 0;

        while (true) {

            switch (prochain()) {

                case un:   if (val <= 2 || (val >= 5 && val <= 7)) {
                               val += 1;
                               pos++;
                           } else {
                               return val;
                           }
                           break;

                case cinq: if (val <= 1) {
                               val = 5 - val;
                               pos++;
                           } else {
                               return val;
                           }
                           break;

                case dix:  if (val == 1) {
                              val = 9;
                              pos++;
                           } else {
                              return val;
                           }
                           break;

                default:  return val;
            }
        }
    };

    var m = chiffre("M", "",  "" );
    var c = chiffre("C", "D", "M");
    var d = chiffre("X", "L", "C");
    var u = chiffre("I", "V", "X");

    if (pos == t.length && pos > 0) {
        return m*1000 + c*100 + d*10 + u;
    } else {
        return -1;
    }
};
```

```
var romain = function (t) {

    var pos = 0;

    var prochain = function () {
        if (pos < t.length) {
            return t.charAt(pos);
        } else {
            return " ";
        }
    };

    var chiffre = function (un, cinq, dix) { … };

    var m = chiffre("M", "",  "" );
    var c = chiffre("C", "D", "M");
    var d = chiffre("X", "L", "C");
    var u = chiffre("I", "V", "X");

    if (pos == t.length && pos > 0) {
        return m*1000 + c*100 + d*10 + u;
    } else {
        return -1;
    }
};
```

```
var chiffre = function (un, cinq, dix) {

    var val = 0;

    while (true) {

        switch (prochain()) {

            case un:   if (val <= 2 || (val >= 5 && val <= 7)) {
                           val += 1;
                           pos++;
                       } else {
                           return val;
                       }
                       break;

            case cinq: if (val <= 1) {
                           val = 5 - val;
                           pos++;
                       } else {
                           return val;
                       }
                       break;

            case dix:  if (val == 1) {
                           val = 9;
                           pos++;
                       } else {
                           return val;
                       }
                       break;

            default:  return val;
        }
    }
};
```