

# 1. ADN et séquences génomiques

- La cellule, atome du vivant
- Au cœur de la cellule, la molécule d'ADN
- L'ADN code l'information génétique
- Qu'est-ce qu'un algorithme ?
- Compter les nucléotides
- Contenu en G-C et A-T des séquences
- Promenade sur l'ADN
- **Changer l'échelle du chemin**
- Prédire l'origine de réplication ?
- Des fenêtres glissantes et recouvrantes

# Oui mais, et la taille de l'écran ?

- **Résolution d'un écran**

- Le nombre de pixels qui peuvent être affichés dans chacune des deux dimensions
- Par exemple : 1024 x 768

- **Problème :**

Comment « faire rentrer » des suites de plusieurs millions, voire milliards, de segments sur un seul et même écran ?

# Oui mais, et la taille de l'écran ?

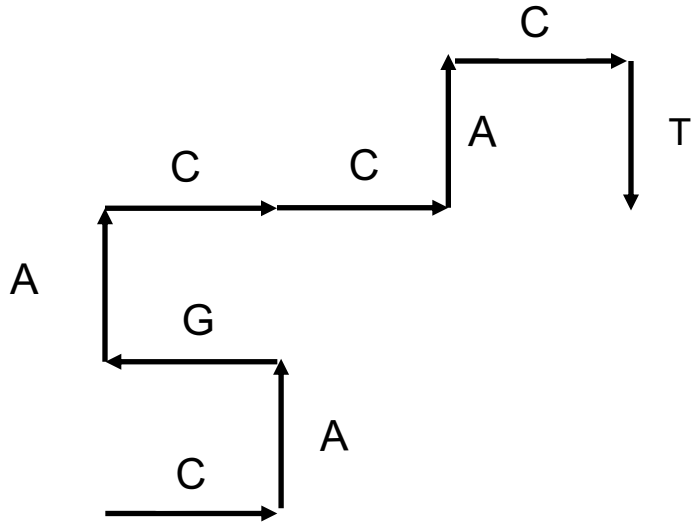
- **Résolution d'un écran**

- Le nombre de pixels qui peuvent être affichés dans chacune des deux dimensions
- Par exemple : 1024 x 768

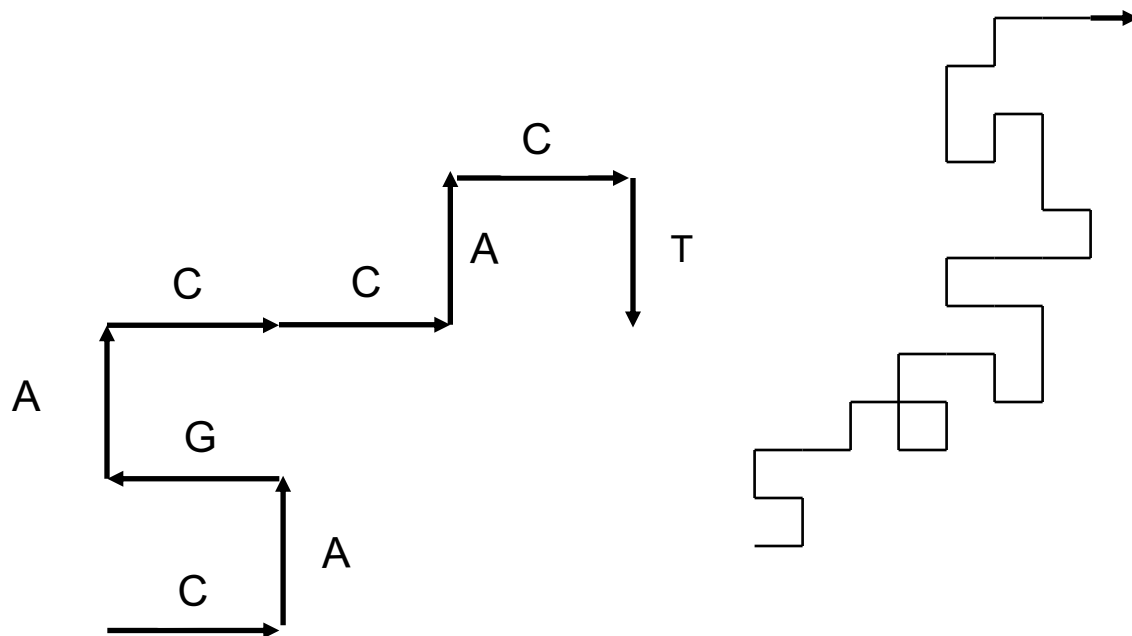
- **Problème :**

Comment « faire rentrer » des suites de plusieurs millions, voire milliards, de segments sur un seul et même écran ?

- La **solution** : changer l'échelle du dessin



**CAGACCACTCAGACCTCAAGGACCCAGAAGTGAACACC...**

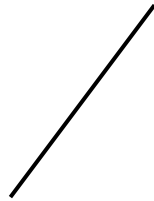


**CAGACCACTCAGACCTCAAGGACCCAGAAGTGAACAC**



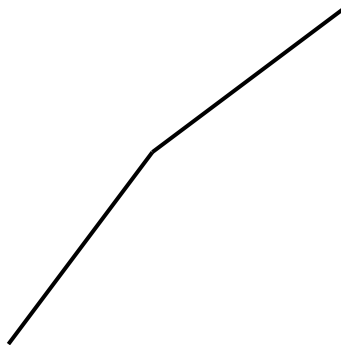
**CAGACCACTCAGACCTCAAGGACCCAGAAAGTGAACAC**

- Calculer nbA, nbC, nbG, nbT dans la fenêtre courante de longueur L
- Calculer les coordonnées de l'extrémité du nouveau segment
- Tracer ce segment
- Avancer la fenêtre le long de la séquence



**CAGACCACTCAGACCTCAAGGACCCAGAAGTGAACAC**

- Calculer nbA, nbC, nbG, nbT dans la fenêtre courante de longueur L
- Calculer les coordonnées de l'extrémité du nouveau segment
- Tracer ce segment
- Avancer la fenêtre le long de la séquence

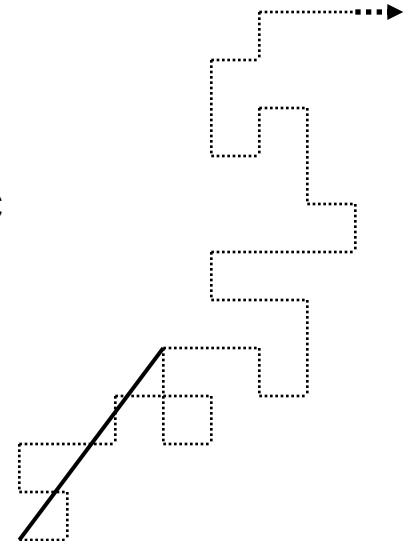
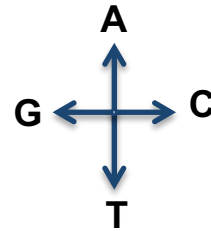




```
L, I, nbA,nbC,nbG,nbT: integer
sequence: character string [1:*]
nbA,nbC,nbG,nbT  $\leftarrow$  0
for I from 1 to L do
  case sequence [I] of
    "A": nbA  $\leftarrow$  nbA + 1
    "C": nbC  $\leftarrow$  nbC + 1
    "G": nbG  $\leftarrow$  nbG + 1
    "T": nbT  $\leftarrow$  nbT + 1
  endcase
endfor
```

```
until InitW > SeqLength
```

**CAGACCACTCAGACCTCAAGGACCCAGAAGTGAACAC**



SeqLength, L, I, InitW, nbA,nbC,nbG,nbT, NbStepsRight, NbStepsUp: **integer**

XEndSegment, YEndSegment, Step: **real**

sequence: **character string** [1:\*

InitW ← 1

**repeat**

    nbA,nbC,nbG,nbT ← 0

**for** I **from** InitW **to** InitW + L - 1 **do**

**case** sequence [I] **of**

            "A": nbA ← nbA + 1

            "C": nbC ← nbC + 1

            "G": nbG ← nbG + 1

            "T": nbT ← nbT + 1

**endcase**

**endfor**

    NbStepsRight ← nbC - nbG

    NbStepsUp ← nbA - nbT

    XEndSegment ← NbStepsRights \* Step

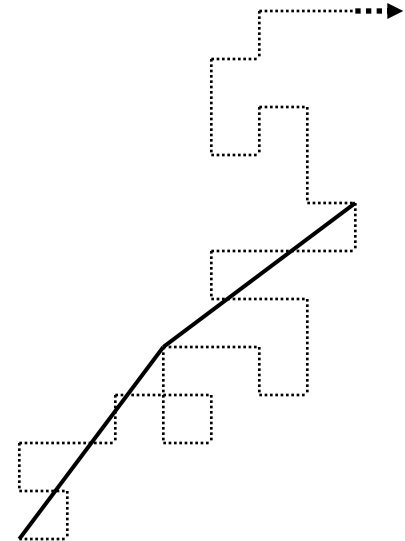
    YEndSegment ← NbStepsUp \* Step

    DrawTill (XEndSegment, YEndSegment)

    InitW ← InitW + L

**until** InitW > SeqLength

CAGACCACTCAGACCTCAAGGACCCAGAAGTGAACAC



SeqLength, L, I, InitW, nbA,nbC,nbG,nbT, NbStepsRight, NbStepsUp: **integer**

XEndSegment, YEndSegment, Step: **real**

sequence: **character string** [1:\*

InitW ← 1

**repeat**

    nbA,nbC,nbG,nbT ← 0

**for** I **from** InitW **to** InitW + L - 1 **do**

**case** sequence [I] **of**

            "A": nbA ← nbA + 1

            "C": nbC ← nbC + 1

            "G": nbG ← nbG + 1

            "T": nbT ← nbT + 1

**endcase**

**endfor**

    NbStepsRight ← nbC - nbG

    NbStepsUp ← nbA - nbT

    XEndSegment ← NbStepsRights \* Step

    YEndSegment ← NbStepsUp \* Step

    DrawTill (XEndSegment, YEndSegment)

    InitW ← InitW + L

**until** InitW > SeqLength

**CAGACCACTCAGACCTCAAGGACCCAGAAGTGAACAC**

