

//

G.A.P.LO: Générateur Automatique de Programmes LOGO.

Le texte suivant est un générateur aléatoire de programme LOGO. Il est donc l'équivalent informatique des G.A.T.O. pour les textes poétiques et des G.A.M.O. pour la musique. Le texte de programmation créé est affiché dans la zone de l'afficheur de texte situé dans la fenêtre de travail. Pour récupérer ce texte, il faut cliquer trois fois sur l'afficheur et sélectionner le texte LOGO produit en le surlignant avec la souris, puis le copier dans le presse-papier. Il ne reste plus qu'à effacer le programme générateur et à coller le texte dans l'éditeur.

Ce texte illustre également une combinaison de caractères utiles pour l'affichage des crochets et des guillemets dans une liste. Ainsi:

```
^^ sera affiché comme un crochet ouvrant [ .  
^^ sera affiché comme un crochet fermant ] .  
|| (Alt Gr droit + 6) sera affiché comme des guillemets " .
```

//

```
efftxt  
donne "consignes [ donne avance baissecrayon fcc fixehasard lèvecrayon op origine recule répète td teste tg ]  
donne "fonctionsnum [ cap hasard pi sourisx sourisy xcoord ycoord ]  
donne "fonctionsliste [ limites? lieu situation ]  
donne "fonctionsbool [ faux surécran? visible? vrai ]  
donne "couleurs [ blanche verte jaune rouge orange tilleul fuchsia sarcelle marron rose violette argent grise noire bleue olive ]  
donne "programme [ eff efftxt accélère déroule baissecrayon ]  
donne "listechosescouleur []  
donne "listechosesnombre []  
donne "crochet 0  
donne "anciennesconsignes [ baissecrayon]
```

```
pour AjouteConsigne :c  
teste (card :anciennesconsignes) = 3  
sivrai donne "anciennesconsignes sp :anciennesconsignes  
donne "anciennesconsignes ph :anciennesconsignes :c  
fin
```

```
pour nomchose  
fixehazard 1 5 donne "l hasard  
fixehazard 1 26 donne "n "  
répète :l [ donne "n mot :n car (96 + hasard) ]  
rends mot "|| :n // || représente " //  
fin
```

```
pour consigne  
donne "c item :consignes hasard  
tantque membre? :c :anciennesconsignes donne "c item :consignes hasard  
si :c <> "op AjouteConsigne :c  
rends :c  
fin
```

```
pour couleur  
fixehazard 1 16  
rends item :couleurs hasard  
fin
```

```
pour indiceboucle  
fixehazard 2 360  
rends hasard  
fin
```

```
pour longueur  
fixehazard 1 15  
rends hasard  
fin
```

```
pour valeur  
fixehazard -1000 1000  
rends hasard  
fin
```

```
pour recherchechosesnombre  
fixehazard 1 card :listechosesnombre
```

```
rends item :listechosesnombre hasard
fin
```

```
pour recherchechosescouleur
fixehazard 1 card :listechosescouleur
rends item :listechosescouleur hasard
fin
```

```
pour recherchefonctionnum
fixehazard 1 7
rends item :fonctionsnum hasard
fin
```

```
pour recherchefonctionbool
fixehazard 1 4
rends item :fonctionsbool hasard
fin
```

```
pour affectation
donne "nom mot" || "a
tantque ( membre? :nom :listechosesnombre ) ou
(membre? :nom :listechosescouleur ) donne "nom nomchose

teste (card :listechosesnombre ) <= ( card :listechosescouleur )
sivrai [
donne "programme ph :programme ph ph "donne :nom valeur
donne "listechosesnombre ph :listechosesnombre :nom
]
sifaux [
donne "programme ph :programme ph ph "donne :nom couleur
donne "listechosescouleur ph :listechosescouleur :nom
]
]
fin
```

```
pour affecte&calculer
donne "programme ph :programme ph "donne" recherchechosenombre
fixehazard 7 9 donne "cas hasard
si (:cas = 8) et ((card :listechosesnombre) = 0) tantque (:cas = 8) donne "cas hasard
si :cas = 7 donne "programme ph ph ph :programme "(" valeur ")"
si :cas = 8 donne "programme ph :programme ph ":" sp sp recherchechosenombre
si :cas = 9 donne "programme ph :programme recherchefonctionnum
fixehazard 1 4
donne "cas hasard
si :cas = 1 donne "programme ph :programme "+"
si :cas = 2 donne "programme ph :programme "-
si :cas = 3 donne "programme ph :programme "*"
si :cas = 4 donne "programme ph :programme "/"
fixehazard 5 9
donne "cas hasard
si (:cas = 8) et ((card :listechosesnombre) = 0) tantque (:cas = 8) donne "cas hasard
si :cas = 5 [
fixehazard 0 1 donne "complexe hasard
si :complexe = 1 [ // expression calculable complexe. //
donne "programme ph :programme "("
donne "cas hasard
teste :cas = 1
sivrai donne "programme ph ph ph :programme "(" valeur ")"
sifaux donne "programme ph :programme ph ":" sp sp recherchechosenombre
fixehazard 1 4
donne "cas hasard
si :cas = 1 donne "programme ph :programme "+"
si :cas = 2 donne "programme ph :programme "-
si :cas = 3 donne "programme ph :programme "*"
si :cas = 4 donne "programme ph :programme "/"
]
donne "programme ph :programme "cos"
fixehazard 0 1
donne "cas hasard
teste :cas = 0
sivrai donne "programme ph ph ph :programme "(" valeur ")"
sifaux donne "programme ph :programme ph ":" sp sp recherchechosenombre
```

```

        si :complexe = 1 donne "programme ph :programme ")
    ]
si :cas = 6 [
    fixehasard 0 1 donne "complexe hasard
    si :complexe = 1 [ // expression calculable complexe. //
        donne "programme ph :programme "("
        donne "cas hasard
        teste :cas = 1
        sivrai donne "programme ph ph ph :programme "(" valeur ")"
        sifaux donne "programme ph :programme ph ":" sp sp recherchechosenombre
        fixehasard 1 4
        donne "cas hasard
        si :cas = 1 donne "programme ph :programme "+"
        si :cas = 2 donne "programme ph :programme "-"
        si :cas = 3 donne "programme ph :programme ""
        si :cas = 4 donne "programme ph :programme "/"
        ]
        donne "programme ph :programme "sin"
        fixehasard 0 1
        donne "cas hasard
        teste :cas = 0
        sivrai donne "programme ph ph ph :programme "(" valeur ")"
        sifaux donne "programme ph :programme ph ":" sp sp recherchechosenombre
        si :complexe = 1 donne "programme ph :programme ")"
    ]
si :cas = 7 donne "programme ph ph ph :programme "(" valeur ")"
si :cas = 8 donne "programme ph :programme ph ":" sp sp recherchechosenombre
si :cas = 9 donne "programme ph :programme recherchefonctionnum
fin

```

```

fixehasard 1 10
répète 5 affectation

```

```

pour construction
fixehasard 2 13
donne "cons consigne
si :cons = "donne [ affectation AjouteConsigne "donne ]

```

```

si :cons = "fcc [
    si ( membre? (:anciennesconsignes) "fcc ) [
        fixehasard 0 1 AjouteConsigne "fcc
        donne "cas hasard
        si (:cas = 1) et (( card :listechosesnombre) = 0) donne "cas 0
        teste :cas = 0 // L'argument est-il une couleur (valeur immédiate) ? //
        sivrai [ donne "programme ph :programme ph "fcc couleur ]
        sifaux [ // L'argument est une chose. //
            donne "programme ph :programme ph "fcc ph ":" sp sp recherchechosescouleur
        ]
    ]
]

```

```

si :cons = "répète [
    fixehasard 0 2
    donne "cas hasard
    si (:cas = 1) et (( card :listechosesnombre) = 0) donne "cas 0
    teste :cas = 0 // L'argument est-il une valeur immédiate ? //
    sivrai donne "programme ph :programme ph ph "répète abs valeur "" // [ //
    sifaux [
        teste :cas = 1
        sivrai donne "programme ph :programme ph ph "répète ph ":" sp sp recherchechosenombre ""
        sifaux donne "programme ph :programme [ répète hasard ]
    ]
        donne "crochet :crochet + 1
        fixehasard 1 3 donne "cas hasard
        répète :cas construction
    teste ( der :programme ) = ""
    sivrai donne "programme sd sd sd :programme
    sifaux donne "programme ph :programme ""
    donne "crochet :crochet - 1
    AjouteConsigne "
]

```

```

si :cons = "teste [
  fixehasard 0 4
  donne "cas hasard AjouteConsigne "teste
  si ( :cas = 1 ) et (( card :listechosesnombre) = 0) donne "cas 0
  teste :cas = 0 // L'argument est-il une valeur immédiate ? //
  sivrai donne "programme ph :programme ph "teste valeur
  sifaux [
    teste :cas = 1
    sivrai donne "programme ph :programme ph "teste ph ": sp sp recherchechosenombre
    sifaux [
      teste :cas = 2
      sivrai donne "programme ph :programme ph "teste "hasard
      sifaux [
        teste :cas = 3
        sivrai donne "programme ph :programme ph "teste recherchefonctionnum
        sifaux donne "programme ph :programme ph "teste recherchefonctionbool
      ]
    ]
  ]
]

```

```

fixehasard 1 6
donne "cas hasard
si :cas = 1 donne "programme ph :programme "<
si :cas = 2 donne "programme ph :programme ">
si :cas = 3 donne "programme ph :programme "<=
si :cas = 4 donne "programme ph :programme ">=
si :cas = 5 donne "programme ph :programme "="
si :cas = 6 donne "programme ph :programme "<>

```

```

fixehasard 0 4
donne "cas hasard AjouteConsigne "teste
si ( :cas = 1 ) et (( card :listechosesnombre) = 0) donne "cas 0
teste :cas = 0 // L'argument est-il une valeur immédiate ? //
sivrai donne "programme ph :programme valeur
sifaux [
  teste :cas = 1
  sivrai donne "programme ph :programme ph ": sp sp recherchechosenombre
  sifaux [
    teste :cas = 2
    sivrai donne "programme ph :programme hasard
    sifaux [
      teste :cas = 3
      sivrai donne "programme ph :programme recherchefonctionnum
      sifaux donne "programme ph :programme recherchefonctionbool
    ]
  ]
]

```

```

donne "programme ph :programme ph "sivrai" ""^
  fixehasard 1 3 donne "h hasard
  répète :h construction
teste (der :programme) = ""^
sivrai donne "programme sd sd :programme
sifaux donne "programme ph :programme ""^
AjouteConsigne "teste
donne "programme ph :programme ph "sifaux" ""^
  fixehasard 1 3 donne "h hasard
  répète :h construction
teste (der :programme) = ""^
sivrai donne "programme sd sd :programme
sifaux donne "programme ph :programme ""^
AjouteConsigne "teste
]

```

```

si :cons = "avance [
  si non (membre? :anciennesconsignes "avance ) [
    fixehasard 0 2
    donne "cas hasard AjouteConsigne "avance
    si ( :cas = 1 ) et (( card :listechosesnombre) = 0) donne "cas 0
    teste :cas = 0 // L'argument est-il une valeur immédiate ? //
  ]
]

```

```

sivrai donne "programme ph :programme ph "avance longueur
sifaux [
  teste :cas = 1
sivrai donne "programme ph :programme ph "avance ph ": sp sp recherchechosenombre
sifaux donne "programme ph :programme ph "avance "hasard
]
]
]

```

```

si :cons = "recale [
  si non (membre? :anciennesconsignes "recale ) [
    fixehasard 0 2
    donne "cas hasard AjouteConsigne "recale
    si ( :cas = 1 ) et (( card :listechosesnombre) = 0) donne "cas 0
    teste :cas = 0 // L'argument est-il une valeur immédiate ? //
    sivrai donne "programme ph :programme ph "recale longueur
    sifaux [
      teste :cas = 1
    sivrai donne "programme ph :programme ph "recale ph ": sp sp recherchechosenombre
    sifaux donne "programme ph :programme ph "recale "hasard
    ]
  ]
]
]

```

```

si :cons = "tg [
  si non (membre? :anciennesconsignes "tg ) [
    fixehasard 0 3
    donne "cas hasard AjouteConsigne "tg
    si ( :cas = 1 ) et (( card :listechosesnombre) = 0) donne "cas 0
    teste :cas = 0 // L'argument est-il une valeur immédiate ? //
    sivrai donne "programme ph :programme ph "tg longueur
    sifaux [
      teste :cas = 1
    sivrai donne "programme ph :programme ph "tg ph ": sp sp recherchechosenombre
    sifaux [
      teste :cas = 2
      sivrai donne "programme ph :programme ph "tg "hasard
      sifaux donne "programme ph :programme ph "tg "compas
    ]
  ]
]
]

```

```

si :cons = "td [
  si non (membre? :anciennesconsignes "td ) [
    fixehasard 0 3
    donne "cas hasard AjouteConsigne "td
    si ( :cas = 1 ) et (( card :listechosesnombre) = 0) donne "cas 0
    teste :cas = 0 // L'argument est-il une valeur immédiate ? //
    sivrai donne "programme ph :programme ph "td longueur
    sifaux [
      teste :cas = 1
    sivrai donne "programme ph :programme ph "td ph ": sp sp recherchechosenombre
    sifaux [
      teste :cas = 2
      sivrai donne "programme ph :programme ph "td "hasard
      sifaux donne "programme ph :programme ph "td "compas
    ]
  ]
]
]

```

```

si :cons = "fixehazard [
  si non (membre? :anciennesconsignes "fixehazard ) [
    AjouteConsigne "fixehazard
    fixehasard 0 2
    donne "cas hasard
    si ( :cas = 1 ) et (( card :listechosesnombre) = 0) donne "cas 0
    teste :cas = 0 // L'argument est-il une valeur immédiate ? //
    sivrai donne "programme ph :programme ph "fixehazard indiceboucle
    sifaux [
      teste :cas = 1
    ]
  ]
]

```

```
sivrai donne "programme ph :programme ph "fixehazard ph ": sp sp recherchechosenombre
sifaux donne "programme ph :programme recherchefonctionnum
```

```
]
```

```
fixehazard 0 2
```

```
donne "cas hasard
```

```
si ( :cas = 1 ) et (( card :listechosesnombre) = 0) donne "cas 0
```

```
teste :cas = 0 // L'argument est-il une valeur immédiate ? //
```

```
sivrai donne "programme ph :programme indiceboucle
```

```
sifaux [
```

```
  teste :cas = 1
```

```
  sivrai donne "programme ph :programme ph ": sp sp recherchechosenombre
```

```
  sifaux donne "programme ph :programme recherchefonctionnum
```

```
]
```

```
]
```

```
]
```

```
si :cons = "op affecte&calcule
```

```
si ((:cons = "lèvecrayon) ou
```

```
(:cons = "baissecrayon) ou
```

```
(:cons = "origine)) [
```

```
si ( non (membre? :anciennesconsignes "lèvecrayon ) et
```

```
non ( membre? :anciennesconsignes "baissecrayon) et
```

```
non ( membre? :anciennesconsignes "origine)) [
```

```
  donne "programme ph :programme :cons
```

```
  AjouteConsigne :cons
```

```
]
```

```
]
```

```
retourne
```

```
fin
```

```
répète 5 construction
```

```
ecl [ Pour placer le programme LOGO créé par cet algorithme, revenir sur la fenêtre de travail et cliquer trois fois sur l'afficheur de  
texte pour copier son contenu dans le presse-papier (à surligner). Reste à coller ensuite ce contenu dans l'éditeur LOGO auparavant  
vidé du présent programme constructeur.]
```

```
ecl " ecl "
```

```
ecl :programme
```