

```

donne "colmax 12 donne "lignemax 12
donne "xmin -(23 * entier (:colmax / 2)) donne "ymin (23* entier (:lignemax /2))
donne "listechoix []
fixehazard 0 2
r p te (:lignemax * :colmax) [
    donne "choix hasard
    si :choix = 0 donne "listechoix ph :listechoix "C
    si :choix = 1 donne "listechoix ph :listechoix "P
    si :choix = 2 donne "listechoix ph :listechoix "F
]

```

```

pour lignecolonne :v
donne "ligne entier (:v / :colmax) +1
si (reste :v :colmax) = 0 donne "ligne :ligne -1
donne "colonne :v - (:colmax * (:ligne -1))
rends ph :ligne :colonne
fin

```

```

eff efftxt d roule ct fcc noire
ecl [ ciseaux: couleur argent]
ecl [ pierre: couleur grise]
ecl [ feuille: couleur blanche]
ecl []
ecl [ La pierre casse les ciseaux. (C<P)]
ecl [ Les ciseaux d coupent la feuille. (F<C)]
ecl [ La feuille enveloppe la pierre. (P<F)]
r p te (:lignemax * :colmax) [
    donne "case lignecolonne boucle
    teste ((:lignemax <=21) et (:colmax <=21))
    sivrai [
        donne "position ph :xmin + ((prem :case)*23) :ymin - ((der :case)*23)
        ellipse [ prem :position der :position 10 10 ]
        si (item :listechoix boucle) = "C colorie :position argent
        si (item :listechoix boucle) = "P colorie :position grise
        si (item :listechoix boucle) = "F colorie :position blanche
    ]
    sifaux [
        si (item :listechoix boucle) = "C fcc argent
        si (item :listechoix boucle) = "P fcc grise
        si (item :listechoix boucle) = "F fcc blanche
    ]
    donne "position ph :xmin + ((prem :case)*2) :ymin - ((der :case)*2)
    point ph (((prem :case)*4) - (:colmax *2)) (((der :case)*4) - (:lignemax *2))
    point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4) - (:lignemax *2))
    point ph (((prem :case)*4)-2) - (:colmax *2)) (((der :case)*4) - (:lignemax *2))
    point ph (((prem :case)*4)-2) - (:colmax *2)) (((der :case)*4)-1) - (:lignemax *2))
    point ph (((prem :case)*4)-2) - (:colmax *2)) (((der :case)*4)-2) - (:lignemax *2))
    point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4)-1) - (:lignemax *2))
    point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4)-2) - (:lignemax *2))
    point ph (((prem :case)*4) - (:colmax *2)) (((der :case)*2)-1) - (:lignemax *2))
    point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4)-2) - (:lignemax *2))
]
]

```

```

fixehazard 1 (:lignemax * :colmax) donne "limite 0 donne "compteur 0
tantque 0<1 [ // C'est un "tant que" infini. //
    donne "compteur :compteur +1
    donne " changes 0
    donne "p1 hasard donne "p2 hasard
    teste (item :listechoix :p1) = "C
    sivrai [
        teste (item :listechoix :p2) = "P
        sivrai [
            donne "p :p1
            transforme :p1 transforme :p1
            remplace :listechoix item :listechoix :p2 :p1
            donne "p1 :p
            donne "case lignecolonne :p
            teste ((:lignemax <=21) et (:colmax <=21))
            sivrai [

```

```

    donne "position ph :xmin + ((prem :case)*23) :ymin - ((der :case)*23)
        colorie :position grise
    ]
    sifaux [
        fcc grise
        donne "position ph :xmin + ((prem :case)*2) :ymin - ((der :case)*2)
        point ph (((prem :case)*4) - (:colmax *2)) (((der :case)*4) - (:lignemax *2))
        point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4) - (:lignemax *2))
        point ph (((prem :case)*4)-2) - (:colmax *2)) (((der :case)*4) - (:lignemax *2))
        point ph (((prem :case)*4)-2) - (:colmax *2)) (((der :case)*4)-1) - (:lignemax *2))
        point ph (((prem :case)*4)-2) - (:colmax *2)) (((der :case)*4)-2) - (:lignemax *2))
        point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4)-1) - (:lignemax *2))
        point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4)-2) - (:lignemax *2))
        point ph (((prem :case)*4) - (:colmax *2)) (((der :case)*2)-1) - (:lignemax *2))
        point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4)-2) - (:lignemax *2))
    ]
        donne "échanges :échanges +1
    ]
    sifaux [
        si (item :listechoix :p2) = "F [
            donne "p :p2
            transforme :p2 transforme :p2
            remplace :listechoix item :listechoix :p1 :p2
            donne "p2 :p
            donne "case lignecolonne :p
            teste (:(lignemax <=21) et (:colmax <=21))
                sivrai [
                    donne "position ph :xmin + ((prem :case)*23) :ymin - ((der :case)*23)
                        colorie :position argent
                    ]
                sifaux [
                    fcc argent
                    donne "position ph :xmin + ((prem :case)*2) :ymin - ((der :case)*2)
                    point ph (((prem :case)*4) - (:colmax *2)) (((der :case)*4) - (:lignemax *2))
                    point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4) - (:lignemax *2))
                    point ph (((prem :case)*4)-2) - (:colmax *2)) (((der :case)*4) - (:lignemax *2))
                    point ph (((prem :case)*4)-2) - (:colmax *2)) (((der :case)*4)-1) - (:lignemax *2))
                    point ph (((prem :case)*4)-2) - (:colmax *2)) (((der :case)*4)-2) - (:lignemax *2))
                    point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4)-1) - (:lignemax *2))
                    point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4)-2) - (:lignemax *2))
                    point ph (((prem :case)*4) - (:colmax *2)) (((der :case)*2)-1) - (:lignemax *2))
                    point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4)-2) - (:lignemax *2))
                ]
                    donne "échanges :échanges +1
                ]
            ]
        ]
    ]
    sifaux [
        teste (item :listechoix :p1) = "P
            sivrai [
                si (item :listechoix :p2) = "F [
                    donne "p :p1
                    transforme :p1 transforme :p1
                    remplace :listechoix item :listechoix :p2 :p1
                    donne "p1 :p
                    donne "case lignecolonne :p
                    teste (:(lignemax <=21) et (:colmax <=21))
                        sivrai [
                            donne "position ph :xmin + ((prem :case)*23) :ymin - ((der :case)*23)
                                colorie :position blanche
                            ]
                        sifaux [
                            fcc blanche
                            donne "position ph :xmin + ((prem :case)*2) :ymin - ((der :case)*2)
                            point ph (((prem :case)*4) - (:colmax *2)) (((der :case)*4) - (:lignemax *2))
                            point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4) - (:lignemax *2))
                            point ph (((prem :case)*4)-2) - (:colmax *2)) (((der :case)*4) - (:lignemax *2))
                            point ph (((prem :case)*4)-2) - (:colmax *2)) (((der :case)*4)-1) - (:lignemax *2))
                            point ph (((prem :case)*4)-2) - (:colmax *2)) (((der :case)*4)-2) - (:lignemax *2))
                            point ph (((prem :case)*4)-1) - (:colmax *2)) (((der :case)*4)-1) - (:lignemax *2))
                        ]
                    ]
                ]
            ]
        ]
    ]

```



```

    donne "limite :limite +1
    si (:limite = (:lignemax * :colmax)) [
si (item :listechoix 1)="P
    ecl ph ph [ La pierre a gagné au terme de ] :compteur - :limite "matches.
si (item :listechoix 1)="C
    ecl ph ph [ Les ciseaux ont gagné au terme de ] :compteur - :limite "matches.
si (item :listechoix 1)="F
    ecl ph ph [ La feuille a gagné au terme de ] :compteur - :limite "matches.
        stop
    ]
    ]
sifaux donne "limite 0
]

```