Algorithme Pendu

**Fonction** principale()

**Effacer** console

**Variable** paysPourTableau[50][100], capitalePourTableau[50][100] **en caractère**

**Variable** pays[100], capitaleADeviner[100], capitales[50][100] **en caractère**

**Variable** indexPays, continent, indice, resultatLecture **en entier**

**Variable** fichierDePays **en fichier**

indice <- 0

fichierPays <- **Ouvrir** fichier « fichierDePays.txt » **en** **lecture**

**Si** fichierPays = NULL

**Afficher** Erreur d’ouverture du fichier fichierDePays.txt

**retourne** -1

**Sinon**

**TantQue LireFichier** fichierDePays, paysPourTableau[indice], capitalePourTableau[indice] <> EOF

Indice <- indice + 1

**FinTantQue**

**FinSi**

**FermerFichier** fichierDePays

**Faire**

continent <- menu()

**Selon** continent

**Cas** 1

indexPays <- genererNombreAleatoire(0, 9)

pays <- paysPourTableau[indexPays]

capitaleADeviner <- capitalePourTableau[indexPays]

**Cas** 2

indexPays <- genererNombreAleatoire(10, 19)

pays <- paysPourTableau[indexPays]

capitaleADeviner <- capitalePourTableau[indexPays]

**Cas** 3

indexPays <- genererNombreAleatoire(20, 29)

pays <- paysPourTableau[indexPays]

capitaleADeviner <- capitalePourTableau[indexPays]

**Cas** 4

indexPays <- genererNombreAleatoire(30, 39)

pays <- paysPourTableau[indexPays]

capitaleADeviner <- capitalePourTableau[indexPays]

**Cas** 5

indexPays <- genererNombreAleatoire(40, 49)

pays <- paysPourTableau[indexPays]

capitaleADeviner <- capitalePourTableau[indexPays]

devinerCapitale(pays, capitaleADeviner, capitales)

**TantQue** continent **<>** 6

**Fin Fonction**

**Fonction** gotoxy (colonne **en entier**, ligne **en entier**)

**Mettre curseur à** colonne, ligne

**FinFonction**

**Fonction** menu() **en entier**

**variable** continent **en entier**

**Faire**

gotoxy(33, 2) **Ecrire** « JEU DU PENDU »

gotoxy(32, 3) **Ecrire** « MENU PRINCIPAL »

gotoxy(14, 8) **Ecrire** « 1. Afrique »

gotoxy(14, 9) **Ecrire** « 2. Amerique »

gotoxy(14, 10) **Ecrire** « 3. Asie »

gotoxy(14, 11) **Ecrire** « 4. Europe »

gotoxy(14, 12) **Ecrire** « 5. Oceanie »

gotoxy(14, 13) **Ecrire** « 6. Quitter »

gotoxy(14, 15) **Ecrire** « Veuillez choisir le continent : »

**lire** continent

**TantQue** continent < 1 **et** continent > 6

**Retourne** continent

**FinFonction**

**Fonction** genererNombreAleatoire(minimum **en entier**, maximum  **en entier**) **en entier**

**Variable** nbrAleatoire **en entier**

nbrAleatoire <- Générer un nombre aléatoire entre minimum et maximum

**Retourne** nbrAleatoire

**FinFonction**

**Fonction** devinerCapitale(pays[100] **en caractère**, capitaleADeviner[100] **en caractère**, capitales[50][100] **en caractère**)

**Variable** compteur **en entier**

**Variable** indice = 0 **en entier**

**Variable** insertionAleatoire **en entier**

**Variable** nombreEssai = 0 **en entier**

**Variable** choixUtilisateur **en entier**

**Variable** listeCapitale[10][100] **en caractère**

**Variable** testEgaliteTableau **en booléen**

**Variable** capitaleTrouvee **en booléen**

testEgaliteTableau <- Vrai

**TantQue** testEgaliteTableau

testEgaliteTableau <- Faux

insertionAleatoire <- genererNombreAleatoire(0, 49)

**Si** capitaleADeviner = capitales[insertionAleatoire]

testEgaliteTableau = Vrai

**FinSi**

**FinTantQue**

listeCapitale[0] <- capitales[insertionAleatoire]

**Pour** compteur **<-** 1 **à** 10

testEgaliteTableau <- Vrai

**TantQue** testEgaliteTableau

testEgaliteTableau <- Faux

insertionAleatoire <- genererNombreAleatoire(0, 49)

**Pour** indice <-0 **à** compteur

**Si** capitaleADeviner = capitale[insertionAleatoire] **ou** listeCapitale[indice] = capitales[insertionAleatoire]

testEgaliteTableau <- True

**FinSi**

**FinPour**

**FinTantQue**

listeCapitale [compteur] <- capitales[insertionAleatoire]

**Fin Pour**

insertionAleatoire <- genererNombreAleatoire(0 et 9)

listeCapitale [insertionAleatoire] <- capitaleADeviner

nombreEssai <- 0

**TantQue** nombreEssaie <2 **et NON** capitaleTrouvee

**Effacer** console

potence(nombreEssai)

**Pour** compteur <- 0 **à** 9

gotoxy(80, 16 + compteur) **Afficher** compteur + 1

gotoxy(82, 16 + compteur) **Afficher** listeCapitale[compteur]

**FinPour**

gotoxy(20, 21) **Afficher** Quelle est la capitale du pays suivant pays

gotoxy(20, 23) **Afficher** Entrez votre choix

gotoxy(41, 23) **Lire** choixUtilisateur

**Si** capitaleADeviner = listeCapitale[choixUtilisateur - 1]

gotoxy(20, 23) **Afficher** Bravo, vous avez gagné !

gotoxy(20, 24) **Afficher** La capitale de ce pays est bien capitaleADeviner

capitaleTrouvee <- Vrai

gotoxy(20, 25) **Afficher** Appuyez sur une touche pour continuer…

**Lire** choixUtilisateur

**Effacer** console

**Sinon**

nombreEssai <- nombreEssai + 1

**FinSi**

**Si** nombreEssai = 2

potence(2)

gotoxy(20, 23) **Afficher** Vous avez perdu !

gotoxy(20, 24) **Afficher** La capitale de ce pays est capitaleADeviner

gotoxy(20, 25) **Afficher** Appuyez sur une touche pour continuer…

**Lire** choixUtilisateur

**Effacer** console

**FinSi**

**FinTantQue**

**Fin Fonction**

**Fonction** basePotence()

gotoxy (27, 2) JEU DU PENDU

gotoxy(26, 4) **Afficher** \*\*\*\*\*\*\*\*\*\*\*\*\*

**Pour** conteur <- 12 **à** 0

gotoxy(26, 5 + conteur) **Afficher** \*

**Mettre curseur à** 22, 18 **Afficher** \*\*\*\*\*\*\*\*\*

**Fonction** potence(etatPotence **en entier**)

**Effacer** console

**Selon** etatPotence

**Cas** 0

basePotence()

**Cas** 1

basePotence()

gotoxy(36, 5) **Afficher** \*\*\*\*\*

gotoxy(36, 6) **Afficher** \* \*

gotoxy(36, 7) **Afficher** \*\*\*\*\*

gotoxy(38, 8) **Afficher** \*

gotoxy(35, 9) **Afficher** \*\*\*\*\*\*\*

gotoxy(35, 10) **Afficher** \* \*

gotoxy(35, 11) **Afficher** \* \*

**Cas** 2

basePotence()

gotoxy(36, 5) **Afficher** \*\*\*\*\*

gotoxy(36, 6) **Afficher** \* \*

gotoxy(36, 7) **Afficher** \*\*\*\*\*

gotoxy(38, 8) **Afficher** \*

gotoxy(35, 9) **Afficher** \*\*\*\*\*\*\*

gotoxy(35, 10) **Afficher** \* \*

gotoxy(35, 11) **Afficher** \* \*

gotoxy(35, 12) **Afficher** \*\*\*\*\*\*\*

gotoxy(35, 13) **Afficher**  \* \*

gotoxy(35, 14) **Afficher** \* \*

**FinFonction**