**BTS INFORMATIQUE DE GESTION**

Option : Développeur d’applications

**EPREUVE E5**

**EPREUVE DE PRATIQUE DES TECHNIQUES INFORMATIQUES**

|  |
| --- |
| Nom et prénom du candidat :  VIGNERON Jean-Baptiste  N° candidat: 0313282587 |

Session 2011

**RECAPITULATIF DES COMPETENCES PRESENTEES**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DOMAINE | | | | INTITULE DES ACTIONS ET STAGES | REMARQUE |
| C1  ETUDIER ET PROPOSER | C2  CONCEVOIR  MODELISER | C3  PRODUIRE  DEVELOPPER | C4  ASSISTER  FORMER |
|  |  | C32 |  | PUISSANCE 4  Jeu de puissance 4 à 2 joueurs. (VB6) | Obligatoire |
|  |  | C33 |  | MACHINE A CAFE  Site web en Silverlight représentant une machine à café (Silverlight) | Obligatoire |
|  |  | C34 |  | GESTION BIBLIOTHEQUE PERSONNELLE  Application WinForms permettant de gérer sa bibliothèque personnelle : livres, CD de musique et films. (C#/MySQL) | Obligatoire |
|  |  | C35 |  | EPSI’TROMBI - TROMBINOSCOPE  Site affichant le trombinoscope des étudiants, des professeurs et des associations de l’école (PHP/MySQL) | Obligatoire |
|  |  | C36  (+ C32) |  | MULTIWALL  Site d’envoi de messages entre utilisateurs reprenant le principe de Twitter ou des murs Facebook. (PHP/MySQL) | Obligatoire |
|  |  | C38  (+ C32) |  | LECTEUR MULTIMEDIA  Lecteur multimédia permettant de lire et contrôler un son ou une vidéo. (Silverlight) | Facultatif |

|  |  |
| --- | --- |
| Académie de Lille | BTS Informatique de Gestion  Option DA |

**EPREUVE PRATIQUE DES TECHNIQUES INFORMATIQUES**

**Fiche de compétences**

Nom et Prénom du Candidat : VIGNERON Jean-Baptiste N°candidat : 0313282587

**Activité**

|  |
| --- |
| Titre de l’activité : PUISSANCE 4  Cadre de l’activé : Idée personnelle  Date de Lancement : Décembre 2009 Fiche n°1 |

**Domaine de compétence principale**

**DA**

**PRODUIRE-DEVELOPPER**

**Compétences professionnelles mises en œuvre**

|  |
| --- |
| * **Compétence principale**   **C32 : Développer selon un algorithme procédural** |

|  |
| --- |
| * **Autres compétences** |

**Liste des travaux**

|  |  |  |
| --- | --- | --- |
| Travaux effectués | Environnement matériel | Logiciels utilisés |
| Réalisation des images  Développement du logiciel | Core 2 Duo (2.1 Ghz)  3 Go de RAM  NVIDIA GeForce 8600M | Visual Basic 6.0  Microsoft Paint |

**Documents présentés**

|  |  |  |
| --- | --- | --- |
| Documents | Type | Observations |
|  |  |  |

**Configuration requise pour la présentation**

|  |  |
| --- | --- |
| Matériels | Intel Pentium 2 ou supérieur  Carte graphique 32 Mo DDR ou supérieur |
| Logiciels | Microsoft Windows 98 ou supérieur  Microsoft Visual Basic 6.0 |

**Puissance 4**

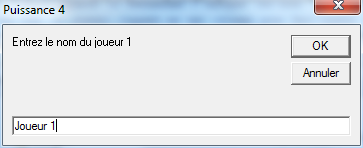
1. **Définition de l’activité**

Ce projet est un jeu de type Puissance 4. Ce logiciel se joue uniquement avec 2 utilisateurs, l’un étant contre l’autre. Il est destiné à tout public, principalement pour les jeunes.

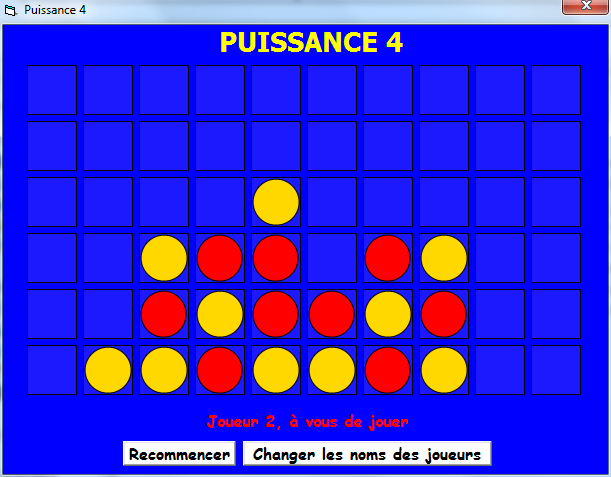
Cette application a été développée avec Microsoft Visual Basic 6 et les images au nombre de 3 : le fond, le pion jaune et le pion rouge ont été réalisés sous Paint.

1. **Spécifications générales :**

Une fois lancé, le programme demande le nom des 2 joueurs. Si le 1er joueur ne rentre pas son nom ou clique sur « Annuler », le programme s’arrête. Si le second joueur ne rentre pas son nom, un message d’alerte apparaît lui demandant d’indiquer son nom.



Une fois les noms rentrés, chacun son tour, les joueurs cliquent sur une colonne pour faire tomber un pion. Le premier joueur réussissant à aligner quatre pions en ligne, en colonne ou en diagonale gagne la partie. Durant la partie ainsi qu’à la fin de celle-ci, il est possible de recommencer la partie ou de changer les noms des joueurs.



1. **Spécifications détaillées :**

* **Structure alternative (If Then Else)**

Cette application contient plusieurs structures alternatives (pour la gestion des joueurs, l’animation du pion qui tombe, la vérification de l’état de la partie (si un joueur a réussi à aligner quatre pions ou si toute la grille est remplie).

* **Structure répétitive (Do While)**

Plusieurs structures répétitives sont présentes permettant l’initialisation de la grille, l’animation du pion qui tombe, les vérifications par ligne, colonne et diagonales, le comptage des pions placés afin de vérifier si toute la grille n’est pas remplie ainsi que le vidage de la grille si les joueurs veulent recommencer une nouvelle partie.

* **Structure de contrôle**

Une structure de contrôle composée d’une structure répétitive associée à une structure alternative permet de vérifier si les joueurs ont bien rentré 2 noms différents au lancement du programme. Dans le cas contraire, la partie ne se lance pas.

|  |
| --- |
| Do While nomJoueurs(0) = "" Or nomJoueurs(1) = ""  nomJoueurs(0) = InputBox("Entrez le nom du joueur 1", "Puissance 4")    If nomJoueurs(0) = "" Then  Unload Partie  Exit Sub  End If    nomJoueurs(1) = InputBox("Entrez le nom du joueur 2", "Puissance 4")    If nomJoueurs(0) = "" Then  MsgBox ("Veuillez entrer un nom pour le joueur 1")  ElseIf nomJoueurs(1) = "" Then  MsgBox ("Veuillez entrer un nom pour le joueur 2")  ElseIf nomJoueurs(0) = nomJoueurs(1) Then  MsgBox ("Veuillez entrer des noms de joueur différents")  nomJoueurs(0) = ""  nomJoueurs(1) = ""  End If  Loop |

* **Structure de données**

La structure de données est matérialisée par un tableau à 2 dimensions représentant la grille du Puissance 4. Celui-ci peut contenir trois valeurs : 0 si aucun pion n’a été placé, 1 si le joueur 1 a placé un pion ou 2 si le joueur 2 a placé un pion.

|  |
| --- |
| Dim tableau(5, 9) As Integer  If tableau(i, j) > 0 Then  compteur = compteur + 1  End If |

* **Appels de procédure et de fonction avec passage de paramètres**

Plusieurs procédures et fonctions sont présentes pour l’animation du pion qui tombe ainsi que les vérifications en ligne, colonne et diagonales.

* + Exemple de procédure :

|  |
| --- |
| Private Sub Animation(ByVal ligne As Integer, ByVal colonne As Integer) |

* + Exemple de fonction :

|  |
| --- |
| Private Function VerificationLigne(ByVal ligne As Integer) As Boolean |

1. **Codage et test :**

J’ai développé cette application avec Visual Basic 6.0 car il permet de réaliser des applications type fenêtre très rapidement en glissant-déposant les éléments depuis la barre d’outils. Visual Basic 6.0 possède également un système d’auto-complétion baptisé IntelliSense permet de développer aisément et plus rapidement.

Visual Basic 6.0 intègre également un débogueur, permettant de placer des points d’arrêt mais ne permettant pas de voir les informations stockées en mémoire. Ce débogueur propose de continuer l’exécution du programme après correction du code ou bien de relancer le programme si nécessaire.

Pour réaliser cette application, j’ai d’abord crée l’interface en reprenant les même couleurs que le vrai jeu Puissance 4, c’est-à-dire bleu, jaune et rouge. J’ai ensuite crée 3 images avec Paint dont une représentant un fond vide, une autre un pion jaune avec un fond bleu et la dernière un pion rouge avec toujours un fond bleu. La seconde étape a consisté a réaliser le code permettant d’ajouter et d’afficher les pions, puis de vérifier si 4 pions étaient alignés. Pour terminer, j’ai réalisé le code permettant de demander et de vérifier les noms de joueurs ainsi que le bouton permettant de recommencer une partie et un autre pour changer les noms des joueurs.

Les tests que j’ai réalisés sont des tests standards. J’ai d’abord testé la vérification de saisie en n’entrant aucune valeur puis en indiquant 2 noms identiques. Une fois le jeu lancé, j’ai posés quelques pions et testé les vérifications par ligne, par colonne et par diagonales. J’ai également rempli la grille de sorte à ce qu’il n’y ait aucun vainqueur afin de voir si un message « Fin. Match nul » s’affichait bien. Entre ces tests, j’ai également testé les boutons permettant de recommencer la partie et de changer les noms des joueurs.

1. **Conclusion :**

Ce projet m’a démontré qu’il était possible de réaliser des jeux simples avec Visual Basic très simplement et rapidement. La réalisation de cette application peut s’avérer plus complexe avec un autre langage comme C ou C++. Les difficultés que j’ai rencontrées sont les vérifications pour les deux diagonales, complexes à mettre en place ainsi que la structure de contrôle imposant l’utilisation d’une structure alternative dans une structure répétitive.