Examen de fin d'études : questions-types

Informatique, I^{re} B

proposition 3 (niveau : plutôt difficile)

Première partie

1. Présentez une version itérative de l'algorithme du *tri par sélection* : code et explications. **[12 p.]**

2. Deux nombres premiers impairs sont appelés *consécutifs* si leur différence est 2.

   Exemples : 17 et 19 sont deux nombres premiers consécutifs.
   67 et 71 ne sont pas deux nombres premiers consécutifs.
   67 et 69 ne sont pas deux nombres premiers consécutifs.

   a) Écrivez une fonction `premier`, dont le résultat est de type *Boolean*, qui vérifie si un nombre naturel donné est premier ou non. **[6 p.]**

   b) Écrivez une procédure qui admet en entrée les deux bornes délimitant un intervalle entier quelconque et qui affiche à l'écran tous les couples de nombres premiers impairs consécutifs contenus dans cet intervalle. **[6 p.]**

3. Déterminez et corrigez les erreurs logiques et syntaxiques de la partie de programme indiquée ci-dessous. L'extrait de programme doit afficher le minimum d'une série de notes (entre 1 et 60) contenues dans le composant `lbListe` de type *TListBox* dans le libellé `lblResultat`. **[6 p.]**

```
MIN:=0;
for I:=0 to lbListe.Items.Count do
  if lbListe.Items[I] < MIN then
    MIN:=StrToInt(lbListe.Items[I]);
lblResultat:='Note minimale: '+IntToStr(MIN);
```

Deuxième partie

Selon les règles du jeu d'échecs, une tour peut battre toute autre figure de l'adversaire qui se trouve dans sa rangée ou sa colonne sous condition qu'il ne se trouve pas d'autre figure entre les deux.

Sur l'échiquier suivant, considérons que les figures a et b appartiennent à l'adversaire, la tour J peut battre toutes les figures a, mais pas les figures b.

			a				
a			J				a
			a				
		b	b			b	

On donne un échiquier carré avec 8×8 cases (représenté par un composant *TStringGrid*) où sont réparties votre tour (lettre J) et les figures de l'adversaire (lettre x). Il s'agit d'écrire une procédure (lancée par un bouton) qui détermine et affiche dans un composant de type *TLabel* le nombre de figures de l'adversaire qui peuvent être battues par votre tour.

S'il n'y a pas de tour ou plusieurs tours, le programme doit afficher un message d'erreur dans le composant de type *TLabel* susmentionné.

Interface graphique : **[8 p.]**

Marche à suivre :

- ⋆ Écrivez d'abord une procédure `checkrook` (*passage par variable*) qui détermine l'emplacement de la tour J. Si l'échiquier ne contient pas de tour, ou s'il en contient plusieurs, `checkrook` donnera des coordonnées négatives et affichera un message d'erreur. **[10 p.]**
- ⋆ Écrivez ensuite deux procédures `checkrow` et `checkcol` qui à partir de l'emplacement de la tour détermine combien de pièces peuvent être battues dans la rangée resp. la colonne de la tour. **[12 p.]**