TD6 : éléments de correction

Exercice 1

Question 1

```
Fonction : Carre
Entrée :        - n : entier
Local :         /
Sortie :        - nCarre : entier
Début
|       Soit nCarre = n * n
|       Retourner nCarre
Fin
```

```
Procédure : Main
Entrée :        void
Local :         - nU : entier
                - nUCarre : entier
Sortie :        void
Début
|       Soit nU = SaisirEntier()
|       Soit nUCarre = Carre(nU)
|       AfficherChaîne("Le carré de {nU} est {nUCarre}.")
Fin
```

Question 2

```
Fonction : Factorielle
Précondition : n ≥ 0
Entrée :        - n : entier
Local :         - i : entier
Sortie :        - nFactorielle : entier
Début
|       Soit nFactorielle = 1
|       Pour i de 2 à n avec un pas de +1 Faire
|       |       Soit nFactorielle = nFactorielle * i
|       Fin Pour
|       Retourner nFactorielle
Fin
```

```
Procédure : Main
Entrée :        void
Local :         - nU : entier
                - nUFactorielle : entier
Sortie :        void
Début
|       Soit nU = SaisirEntier()
|       // test de la précondition
|       Si nU ≥ 0 Alors
|       |       Soit nUFactorielle = Factorielle(nU)
|       |       AfficherChaîne("La factorielle de {nU} est {nUFactorielle}.")
|       Sinon
|       |       AfficherChaîne("Valeur non autorisée {nU} < 0.")
|       Fin Si
Fin
```

Question 3

```
Fonction : Div10
Entrée :        - n : réel
Local :         /
Sortie :        - nDiv10 : réel
Début
|       Soit nDiv10 = n / 10
|       Retourner nDiv10
Fin
```

```
Procédure : Main
Entrée :        void
Local :         - nU : réel
                - nUDiv10 : réel
Sortie :        void
Début
|       Soit nU = SaisirRéel()
|       Soit nUDiv10 = Div10(nU)
|       AfficherChaîne("{nU} / 10 = {nUDiv10}.")
Fin
```

Question 4

```
Procédure : Main (menu des opérations Carre, Factorielle et Div10)
Entrée :        void
Local :         - nombre : réel
                - operation : entier
                - resultat : réel ou entier (selon le calcul)
Sortie :        void
Début
|       Soit nombre = SaisirRéel()
|       Soit operation = SaisirEntier()
|       Si operation == 1 Alors
|       |       Soit resultat = Carre(ConversionEntier(nombre))
|       |       AfficherChaîne("{nombre}^2 = {resultat}.")
|       Sinon Si operation == 2 Alors
|       |       Soit resultat = Factorielle(ConversionEntier(nombre))
|       |       AfficherChaîne("{nombre}! = {resultat}.")
|       Sinon Si operation == 3 Alors
|       |       Soit resultat = Div10(nombre)
|       |       AfficherChaîne("{nombre} / 10 = {resultat}.")
|       Fin Si
Fin
```

Question 5

```
Fonction : Demander10EntiersEtRetournerMinimum
Entrée :        void
Local :         - nombre : entier
                - i : entier
Sortie :        - minimum : entier
Début
|       Soit minimum = SaisirEntier()
|       Pour i de 1 à 9 avec un pas de +1 Faire
|       |       Soit nombre = SaisirEntier()
|       |       Si nombre < minimum Alors
|       |       |       Soit minimum = nombre
|       |       Fin Si
|       Fin Pour
|       Retourner minimum
Fin
```

```
Procédure : Main
Entrée :        void
Local :         - min : entier
Sortie :        void
Début
|       Soit min = Demander10EntiersEtRetournerMinimum()
|       AfficherChaîne("Le minimum des 10 entiers est {min}.")
Fin
```

Question 6

```
Procédure : TableMult
Entrée :        - base : entier
Local :         - i : entier
Sortie :        void
Début
|       Pour i de 0 à 10 avec un pas de +1 Faire
|       |       AfficherChaîne("{base} * {i} = {base * i}.")
|       Fin Pour
Fin
```

```
Procédure : Main
Entrée :        void
Local :         - baseU : entier
Sortie :        void
Début
|       Soit baseU = SaisirEntier()
|       TableMult(baseU)
Fin
```