Spécification des modules Exercice et Correction

Question de type P

Test

domaine
test

fonctions de construction
creer_test : string x string → test

fonctions d'accès
get_in: test → string
get_out: test → string

Liste de tests

domaine
test_liste

constants
test_liste_vide

fonctions de construction
cons_test_liste : test x test_liste → test_liste

fonctions de test
est_test_liste_vide: test_liste → bool

fonctions d'accès
tete_test_liste: {t: test_listel not(est_test_liste_vide(t))} → test
queue_test_liste: {t: test_listel not(est_test_liste_vide(t))} → test_liste

---

Question de type P

domaine
question_p

fonctions de construction

cons_question_p: string x test_liste x int → question_p

fonctions d'accès

get_tests_question_p: question_p → test_liste
get_enonce_question_p: question_p → string
get_note_question_p: question_p → int

Exercice de type P

domaine
exercice_p

constants
exercice_p_vide

fonctions de construction
cons_exercice_p : question_p x exercice_p → exercice_p

fonctions de test
est_exercice_p_vide: exercice_p → bool

fonctions d'accès
tete_exercice_p: {e: exercice_pl not(est_exercice_p_vide(e))} → question_p
queue_exercice_p: {e: exercice_pl not(est_exercice_p_vide(e))} → exercice_p

---

Correction d'une question de type P

passer_test : fun x test → bool

```
passer_test(p,t){
        return (p get_in(t)) ==  get_out(t);
}
```

passer_tests : fun x test_liste → bool

```
passer_tests(p, tl){
        if(est_test_liste_vide(tl)){
                return false;
        } else {
                if(passer_test(fun p x tete_test_liste(tl))){
                        return passer_tests(fun p x queue(tl));
                } else {
                        false;
                }
        }
}
```

corriger_question_p : question_p x fun → int

```
corriger_question_p(q,p){
        if(passer_tests(p x get_tests(q)){
                return get_note_question_p(q);
        } else {
                return 0;
        }
}
```

Question de type QCM

Reponse

domaine
reponse_qcm

fonctions de construction
cons_reponse_qcm: string x bool → reponse_qcm

fonctions de test
est_correct_reponse_qcm: reponse_qcm r → bool
compare_reponse_qcm_string: reponse_qcm x string → bool

fonctions d'accès
get_contenu: reponse_qcm → string

Liste de réponses

domaine
reponse_qcm_liste

constants
reponse_qcm_liste_vide

fonctions de construction
cons_reponse_qcm_liste : reponse_qcm x reponse_qcm_liste → reponse_qcm_liste

fonctions de test
est_reponse_qcm_liste_vide: reponse_qcm_liste → bool

fonctions d'accès
tete_reponse_qcm_liste:{rl: reponse_qcm_liste | not(est_reponse_qcm_liste_vide(rl))} → reponse_qcm
queue_reponse_qcm_liste:{rl: reponse_qcm_liste | not(est_reponse_qcm_liste_vide(rl))} → reponse_qcm_liste

Question de type QCM

domaine
question_qcm

fonctions de construction
cons_question_qcm: string x reponse_qcm_list x int → question_qcm

fonctions d'accès
get_reponses_question_qcm: question_qcm → reponse_qcm_list
get_enonce_question_qcm: question_qcm → string
get_note_question_qcm: question_qcm → int

Exercice de type QCM

domaine
exercice_qcm

constants
exercice_qcm_vide

fonctions de construction
cons_exercice_qcm : question_qcm x exercice_qcm → exercice_qcm

fonctions de test
est_exercice_qcm_vide: exercice_qcm → bool

fonctions d'accès
tete_exercice_qcm: {e: exercice_qcml not(est_exercice_p_vide(e))} → question_qcm
queue_exercice_qcm: {e: exercice_qcml not(est_exercice_p_vide(e))} → exercice_qcm

Correction d'une question de type QCM

corriger_qcm_aux : reponse_qcm_liste x string → bool

```
corriger_qcm_aux(l, s){
        if(est_reponse_qcm_liste_vide(l)){
                return false;
        } else {
          if(compare_reponse_qcm_string(tete_reponse_qcm_liste(l) x s) &&
est_correct_reponse_qcm(tete_reponse_qcm_liste(l))){
                return true;
```

```
            } else {
                corriger_qcm_aux(queue_reponse_qcm_liste(l) x s)
            }
        }
}

corriger_qcm : question_qcm x string → int

corriger_qcm(q, r){
        if(corriger_qcm_aux(get_reponses_question_qcm(q), r)){
                return get_note_question_qcm(q);
        } else {
                return 0;
        }
}
```

Question de type QRF

Reponse

domaine
reponse_qrf

fonctions de construction
cons_reponse_qrf: string→ reponse

fonctions d'accès
get_contenu:{r:reponse_qrf} → string

fonctions de test
compare_reponse_qrf_string:{reponse_qrf x string} → bool

Liste de réponses

domaine
reponse_qrf_liste

constants
reponse_qrf_liste_vide

fonctions de construction
cons_reponse_qrf_liste : reponse_qrf x reponse_qrf_liste → reponse_qrf_liste

fonctions de test
est_reponse_qrf_liste_vide: reponse_qrf_liste → bool

fonctions d'accès
tete_reponse_qrf_liste:{rl: reponse_qrf_liste | not(est_liste_vide(rl))} → reponse_qrf
queue_reponse_qrf_liste:{rl: reponse_qrf_liste | not(est_liste_vide(rl))} → reponse_qrf_liste

---

Question de type QRF

domaine
question_qrf

fonctions de construction
cons_question_qrf: string x reponse_qrf_list x int → question_qrf

fonctions d'accès
get_reponses_question_qrf:{qrf:question_qrf} → reponse_qrf_list
get_enonce_question_qrf:{qrf:question_qrf} → string
get_note_question_qrf:{qrf: question_qrf} → int

Exercice de type QRF

domaine
exercice_qrf

constants
exercice_qrf_vide

fonctions de construction
cons_exercice_qrf : question_qrf x exercice_qrf → exercice_qrf

fonctions de test
est_exercice_qrf_vide: exercice_qrf → bool

fonctions d'accès
tete_exercice_qrf: {e: exercice_qrfl not(est_exercice_p_vide(e))} → question_qrf
queue_exercice_qrf: {e: exercice_qrfl not(est_exercice_p_vide(e))} → exercice_qrf

---

Correction d'une question de type QRF

corriger_qrf_aux : reponse_qrf_liste x string → bool

```
corriger_qrf_aux(l, s){
        if(est_reponse_qrf_liste_vide(l)){
                return false;
        } else {
          if(compare_reponse_qrf_string(tete_reponse_qrf_liste(l) x s)){
                return true;
```

```
            } else {
                corriger_qrf_aux(queue_liste(l) x s)
            }
        }
}

corriger_qrf : question_qrf x string) → int

corriger_qrf(q, r){
        if(corriger_qrf_aux(get_reponses_question_qrf(q), r)){
                return get_note_question_qrf(q);
        } else {
                return 0;
        }
}
```

Question de type L

Question de type L

domaine
question_l

fonctions de construction
cons_l_question: string x int→ question_l

fonctions d'accès
get_enonce:{l:question_l} → string
get_note:{l: question_l} → int

Exercice de type L

domaine
exercice_l

constants
exercice_l_vide

fonctions de construction
cons_exercice_l : question_l x exercice_l → exercice_l

fonctions de test
est_exercice_l_vide: exercice_l → bool

fonctions d'accès

tete_exercice_l: {e: exercice_ll not(est_exercice_p_vide(e))} → question_l
queue_exercice_l: {e: exercice_ll not(est_exercice_p_vide(e))} → exercice_l