```
 1:
 2: int i;
 3: void clignoter (){
 4: for (i=1; i<6; i++)
 5:         {//PORTB.RB1=~PORTB.RB1;
 6:         //delay_ms(500);
 7:         PORTB.RB1 =1;
 8:         delay_ms(500);
 9:         PORTB.RB1 =0;
10:         delay_ms(500);
11:           }
12: }
13: void interrupt()
14: {     // routine d'interruption : conséquences du dé
    éclenchement des interruptions
15:     if ((INTCON.INTE) && (INTCON.INTF))   //le bouto
    on réalisateur a interrompu
16:     {
17:     clignoter();
18:     INTCON.INTF = 0;} //on remet à 0 le drapeau(flag
    g) du bouton réalisateur
19:
20:     if ((INTCON.RBIE) && (INTCON.RBIF))     //Un des
    s candidats a interrompu
21:     {
22:     if (!PORTB.B4) {porta.B3=1;   delay_ms (1000);
    }                                                   //
    dantes pendant 1s
23:     if (!PORTB.B5) {porta.B2=1;   delay_ms (1000);}
                                                        //
    dantes pendant 1s
24:     if (!PORTB.B6) {porta.B1=1; delay_ms (1000);}
                                                        //
    dantes pendant 1s
25:     if (!PORTB.B7) {porta.B0=1; delay_ms (1000);}
26:     INTCON.RBIF = 0;   //on remet à 0 le drapeau (fl
    lag) des boutons des candidats
27:     }
```

```
28:
29: }
30:
31: void main()   //programme principal
32: {
33: // configuration
34: TRISA = 0b0000000;  // port A : configuration en sor
    rties (les LEDs des candidats)
35: TRISB= 0b11110001;  // port B : configuration en ent
    trées (boutons: RB0= bouton réalisateur, RB4,RB5,R
    6= boutons des candidats)
36: //configuration des interruptions:
37: //autoriser bouton réalisateur(source d'interrupti
    n RB0) et RB4-RB5-RB6 (boutons candidats)(la sourc
     d'interruption PORTB)
38: INTCON = 0b10011000;   // bit 7 (GIE) = 1 : autorisa
    ation globale des interruptions
39:                        // bit 4 (INTE) = 1 : autoris
    sation de l'interruption RB0
40:                        // bit 3 (RBIE)= 1 :  autoris
    sation de l'interruption RB4-7
41: //aussi on doit utiliser les résistances de rappel
    internes du PIC pour faire fonctionner les boutons
    connectés au portb  ( donc bit 7 du registre OPTIO
    _REG=0)
42: //Pour la source d'interruption RB0, on doit préci
    er si l'interruption est déclenchée lors du front
    ontant ou du front descendant (bit 6 INTEDG)
43: //puisque dans notre TP, l'interruption sur RB0 se
    déclenche lors de l'appui sur le bouton et puisque
    ce bouton est connecté à la masse, donc l'appui co
    respond à un front descendant
44: //c'est pourquoi on met le bit 6 (INTEDG) du regis
    re OPTION_REG à 0 (front descendant)
45: OPTION_REG = 0b0000000; // avec résistances pull-up
     / front descendant
46: //initialisation
47: PORTA=0b00000000;      //toutes les lampes sont étei
    intes
```

```
48: //boucle infinie du traitement effectué s'il n'y a pas d'interruptions
49: do
50:     {   PORTA=0;
51:         PORTB.B1 = 0;
52:       // les lampes sont éteintes en attendant l'appui sur le bouton RB0 ou l'appui sur les boutons R4-RB7
53:     } while(1);
54: }
```