

```

1  //-----
2  // TP 5 exercice 1
3  #include <stdio.h>
4  #include <stdlib.h>
5  #include <time.h>
6  #define N 10
7
8  int main (void) {
9      int i,neg;
10     int A[N], B[N];
11
12     srand(time(0));
13
14     //remplissage des tableaux
15     for (i=0;i<N;i++) {
16         A[i]=i;
17         B[i]=rand()%10;
18     }
19
20     //affichage des tableaux
21     printf("\nA : ");
22     for (i=0;i<N;i++) {
23         printf("%2d",A[i]);
24     }
25     printf("\nB : ");
26     for (i=0;i<N;i++) {
27         printf("%2d",B[i]);
28     }
29
30     //comparaison des 2 tableaux element par element
31     for (neg=0,i=0;i<N;i++) {
32         if (A[i]==B[i])
33             neg++;
34     }
35
36     //affichage
37     printf("\nil y a %d case(s) identique(s)\n",neg);
38
39     return 0;
40 }
41
42 //-----
43 // TP 5 Exercice 2
44 #include <stdio.h>
45 #define NMAX 10
46
47 int main (void) {
48     int i,max,tab[NMAX];
49
50     //saisie du tableau
51     printf("entrez les valeurs : ");
52     for (i=0;i<NMAX;i++)
53         scanf("%d",&tab[i]);
54
55     //affichage dans l'ordre inverse
56     for (i=NMAX-1;i>=0;i--)
57         printf("%d ",tab[i]);
58
59     //initialisation du max
60     max=tab[0];
61
62     //recherche dans le reste du tableau
63     for (i=1;i<NMAX;i++) {
64         if (tab[i]>max)
65             max=tab[i];
66     }
67
68     //affichage du max
69     printf("\nle maximum est %d\n",max);
70
71     return 0;
72 }
73

```

```

74  //-----
75  // TP 5 Exercice 3
76  #include<stdio.h>
77  #include <string.h>
78  #define NMAX 128
79
80  int letmin(char); //fonction indiquant si le caractere passe en argument est une
    lettre minuscule
81
82  int main (void) {
83      int i,l,n;
84      char chaine[NMAX];
85
86      printf("entrez une chaine de caracteres : ");
87      gets(chaine);
88
89      //on affiche la chaine
90      puts(chaine);
91
92      //longueur de la chaine avec fonction strlen
93      l=strlen(chaine);
94      printf("votre chaine contient %d caracteres\n",l);
95
96      //longueur de la chaine sans fonction strlen
97      for (i=0;i<NMAX;i++)          //on parcourt le tableau
98          if (chaine[i]=='\0')      //on sort de la boucle des qu'on voit
99              break;                //le caractere de fin de chaine
100      printf("votre chaine contient %d caracteres\n",i);
101
102      //nombre de lettres minuscules
103      n=0;
104      for (i=0;i<l;i++)
105          if (letmin(chaine[i]))
106              n++;
107
108      printf("votre chaine contient %d lettres minuscules\n",n);
109
110      return 0;
111  }
112
113  int letmin(char c)
114  {
115      if('a'<=c && c<='z')
116          return 1;
117      else
118          return 0;
119  }
120
121  //-----
122  // TP 5 Exercice 4
123  #include <stdio.h>
124  #define NMAX 20
125
126  //prototypes des fonctions
127  void LirTab(int[],int); //fonction qui remplit le tableau
128  void AffTab(int[],int); //fonction qui affiche le talbeau
129  float MoyTab(int[],int); //fonction qui renvoie la moyenne du tableau
130  int PosTab(int[],int); //fonction qui renvoie le nombre d'elements positifs du
    tableau
131  int ImaxTab(int[],int); //fonction qui renvoie l'indice du maximum du tableau
132  int InTab(int[],int,int); //fonction qui indique si une valeur donnée est présente
    dans le tableau
133
134  int main (void) {
135      int imax,n,npos,x,present,tab[NMAX];
136      float moyenne;
137
138      //saisie du tableau
139      do {
140          printf("entrez la taille du tableau (entre 1 et %d) : ",NMAX);
141          scanf("%d",&n);
142      } while (n<1 || n>NMAX);
143

```

```

144     LirTab(tab,n);
145     AffTab(tab,n);
146     moyenne=MoyTab(tab,n);
147     npos=PosTab(tab,n);
148     imax=ImaxTab(tab,n);
149
150     printf("entrez la valeur recherchee : ");
151     scanf("%d",&x);
152     present=InTab(tab,n,x);
153
154     //affichage
155     printf("la moyenne est %f\n",moyenne);
156     printf("il y a %d elements strictement positifs\n",npos);
157     printf("le max vaut %d, c'est l'element %d du tableau\n",tab[imax],imax+1);
158     if (present)
159         printf("la valeur %d est presente dans le tableau\n",x);
160     else
161         printf("la valeur %d n'est pas presente dans le tableau\n",x);
162
163     return 0;
164 }
165
166 //fonction remplissant le tableau tab de taille n
167 void LirTab(int tab[],int n)
168 {
169     int i;
170     printf("entrez les %d valeurs :\n",n);
171     for (i=0;i<n;i++)
172         scanf("%d",&tab[i]);
173 }
174
175 //fonction affichant le tableau tab de taille n
176 void AffTab(int tab[],int n)
177 {
178     int i;
179     printf("[");
180     for (i=0;i<n;i++)
181         printf("%d ",tab[i]);
182     printf("\b]\n");
183 }
184
185 //fonction renvoyant la moyenne du tableau tab de taille n
186 float MoyTab(int tab[],int n)
187 {
188     int i,somme;
189     for (somme=i=0;i<n;i++) {
190         somme+=tab[i];
191     }
192     return (1.*somme)/n; //on force la division réelle
193 }
194 //fonction renvoyant le nombre de valeurs strictement positives
195 //dans le tableau tab de taille n
196 int PosTab(int tab[],int n)
197 {
198     int i, npos=0;
199     for (i=0;i<n;i++)
200         if (tab[i]>0)
201             npos++;
202     return npos;
203 }
204
205 //fonction renvoyant l'indice du maximum du tableau tab de taille n
206 int ImaxTab(int tab[],int n)
207 {
208     int i,imax;
209     imax=0;
210     for (i=0;i<n;i++) {
211         if (tab[i]>tab[imax]) {
212             imax=i;
213         }
214     }
215     return imax;
216 }

```

```

217
218 //fonction indiquant si la valeur x est presente
219 //dans le tableau tab de taille n
220 int InTab(int tab[],int n,int x)
221 {
222     int i;
223     for (i=0;i<n;i++) {
224         if (tab[i]==x) {
225             return 1;
226         }
227     }
228     return 0; //on a parcouru tout le tableau et
229 }           //on n'a jamais trouve la valeur cherchee
230
231
232
233
234 //-----
235 // TP 5 Exercice 5
236
237 #include<stdio.h>
238 #include <string.h>
239 #include <stdlib.h>
240
241 #define NMAX 128
242
243 int main (void) {
244     int n,i;
245     char radical[NMAX],verbe[NMAX],pc;
246
247     printf("entrez un verbe du premier groupe : ");
248     gets(verbe);
249
250     //on verifie qu'il s'agit d'un verbe du 1er groupe
251     n=strlen(verbe);
252     if ((verbe[n-1]!='r') || (verbe[n-2]!='e') || !strcmp(verbe,"aller")) {
253         printf("vous n'avez pas entre un verbe du 1er groupe, au revoir\n");
254         exit(0);
255     }
256
257     strcpy(radical,verbe);
258     radical[n-2]='\0'; //on supprime la terminaison 'er' du verbe
259
260     printf("votre verbe conjugue donne :\n");
261     printf("je %se\n",radical);
262     printf("tu %ses\n",radical);
263     printf("il %se\n",radical);
264     printf("nous %sons\n",radical);
265     printf("vous %sez\n",radical);
266     printf("ils %sent\n",radical);
267
268     return 0;
269 }
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289

```

```

290 //-----
291 // TP 5 Exercice 6
292
293 #include <stdio.h>
294 #include <time.h>
295 #include <stdlib.h>
296 #define N 101
297
298 //prototypes des fonctions
299 int de_elec(int);
300
301 //programme principal
302 int main(void)
303 {
304     int i,n,p,T[N],p2;
305
306     srand(time(0)); //initialisation des nombres aleatoires
307
308     do {
309         printf("nombre de faces de votre de (compris entre 1 et %d) : ",N-1);
310         scanf("%d",&n);
311     } while (n<1 || n>=N);
312
313     do {
314         printf("nombre total de lancers : ");
315         scanf("%d",&p);
316         if (p<=0)
317             printf("rentrez une valeur positive !!\n\n");
318     } while (p<=0);
319
320     //initialisation des compteurs
321     for (i=1;i<=n;i++) //la case T[i] contiendra le nombre de fois
322         T[i]=0; //ou le de est tombe sur la face i
323
324     //boucle sur les lances de dés
325     for (i=0;i<p;i++)
326     {
327         T[de_elec(n)]++;
328     }
329
330     //affichage des resultats
331     printf("Sur %d lances de de :\n",p);
332     for (i=1;i<=n;i++)
333         printf("nombre de %d : %6d  (%2.2f%%)\n",i,T[i],100.*T[i]/p);
334
335     //verification finale
336     p2=0;
337     for (i=1;i<=n;i++)
338         p2+=T[i];
339     if (p2!=p)
340         printf("des des sont apparus ou ont disparus !!\n");
341
342     return 0;
343 }
344
345 //definition des fonctions
346 int de_elec(int a) {
347     return (1+rand()%a);
348 }
349
350
351
352
353
354
355 //-----
356 // TP 5 Exercice 7
357
358 #include <stdio.h>
359 #include <stdlib.h>
360 #include <time.h>
361 #define NMAX 20
362

```

```

363 //prototypes des fonctions
364 int Aleat(int min, int max);
365 void RandTab(int T[], int n, int min, int max);
366 void AffTab(int T[],int n);
367 void TriTab(int T[], int n);
368
369 int main(void)
370 {
371     int n,min,max,Tab[NMAX];
372
373     do {
374         printf("entrez la taille du tableau (entre 1 et %d) : ",NMAX);
375         scanf("%d",&n);
376     } while (n<1 || n>NMAX);
377
378     printf("valeur min : ");
379     scanf("%d",&min);
380     do {
381         printf("valeur max : ");
382         scanf("%d",&max);
383     } while (max < min);
384
385     RandTab(Tab,n,min,max);
386     printf("Voici le tableau : \n");
387     AffTab(Tab,n);
388     TriTab(Tab,n);
389     printf("Voici le tableau trie : \n");
390     AffTab(Tab,n);
391
392     return 0;
393 }
394
395 int Aleat(int min, int max)
396 {
397     static int i=0;
398     if (i==0)
399         srand(time(i++));
400
401     return min+rand()%(max-min+1);
402 }
403
404 void RandTab(int T[], int n, int min, int max)
405 {
406     int i;
407     for (i=0;i<n;i++)
408         T[i]=Aleat(min,max);
409 }
410
411 void AffTab(int T[],int n)
412 {
413     int i;
414     printf("\n[");
415     for (i=0;i<n;i++)
416         printf("%d ",T[i]);
417     printf("\b]\n");
418 }
419
420 void TriTab(int T[], int n)
421 {
422     int i,j,jmin,temp;
423     for (i=0;i<n-1;i++) {
424         //on cherche le minium entre la case i et la fin du tableau
425         jmin=i;
426         for(j=i+1;j<n;j++)
427             if (T[j]<T[jmin])
428                 jmin=j;
429         //on permute T[i] et ce minimum
430         temp=T[i];
431         T[i]=T[jmin];
432         T[jmin]=temp;
433     }
434 }
435

```

```

436 //-----
437 // TP 5 Exercice 8
438
439 #include <stdio.h>
440 #include <stdlib.h>
441 #include <time.h>
442 #define NMAX 20
443 #define NR 10
444
445 //prototypes des fonctions
446 void LirTab(int[],int); //fonction qui remplit le tableau
447 void AffTab(int[],int); //fonction qui affiche le tableau
448 float MoyTab(int[],int); //fonction qui renvoie la moyenne du tableau
449 int PosTab(int[],int); //fonction qui renvoie le nombre d'elements positifs
    du tableau
450 int ImaxTab(int[],int); //fonction qui renvoie l'indice du maximum du tableau
451 int InTab(int[],int,int); //fonction qui indique si une valeur donnée est
    présente dans le tableau
452 int SuppValTab(int[],int,int); //fonction supprimant une valeur donnée dans le
    tableau
453 int MedTab(int[],int); //fonction renvoyant la mediane du tableau
454
455 int main (void) {
456     int imax,n,npos,x,present,tab[NMAX],med;
457     float moyenne;
458
459     //saisie du tableau
460     do {
461         printf("entrez la taille du tableau (entre 1 et %d) : ",NMAX);
462         scanf("%d",&n);
463     } while (n<1 || n>NMAX);
464
465     LirTab(tab,n);
466     AffTab(tab,n);
467     moyenne=MoyTab(tab,n);
468     npos=PosTab(tab,n);
469     imax=ImaxTab(tab,n);
470
471     printf("entrez la valeur recherchee : ");
472     scanf("%d",&x);
473     present=InTab(tab,n,x);
474
475     //affichage
476     printf("la moyenne est %f\n",moyenne);
477     printf("il y a %d elements strictement positifs\n",npos);
478     printf("le max vaut %d, c'est l'element %d du tableau\n",tab[imax],imax+1);
479     if (present)
480         printf("la valeur %d est presente dans le tableau\n",x);
481     else
482         printf("la valeur %d n'est pas presente dans le tableau\n",x);
483
484     //calcul et affichage de la mediane
485     med=MedTab(tab,n);
486     printf("la mediane est %d\n",med);
487
488     //Suppression d'une valeur
489     printf("quelle valeur voulez vous supprimer du tableau : ");
490     scanf("%d",&x);
491     n=SuppValTab(tab,n,x);
492     printf("Voici le nouveau tableau : \n");
493     AffTab(tab,n);
494
495     return 0;
496 }
497
498
499
500
501
502
503
504
505

```

```

506 //fonction remplissant le tableau tab de taille n
507 void LirTab(int tab[],int n)
508 {
509     int i;
510     //printf("entrez les %d valeurs :\n",n);
511     srand(time(0));
512     for (i=0;i<n;i++)
513         //scanf("%d",&tab[i]);
514         tab[i]=rand()%(2*NR+1)-NR; //on remplit le tableau avec des valeurs
515                                     //aleatoires entre -NR et NR
516 }
517
518 //fonction affichant le tableau tab de taille n
519 void AffTab(int tab[],int n)
520 {
521     int i;
522     printf("[");
523     for (i=0;i<n;i++)
524         printf("%d ",tab[i]);
525     printf("\b]\n");
526 }
527
528 //fonction renvoyant la moyenne du tableau tab de taille n
529 float MoyTab(int tab[],int n)
530 {
531     int i,somme;
532     for (somme=i=0;i<n;i++) {
533         somme+=tab[i];
534     }
535     return (1.*somme)/n; //on force la division réelle
536 }
537
538 //fonction renvoyant le nombre de valeurs strictement positives
539 //dans le tableau tab de taille n
540 int PosTab(int tab[],int n)
541 {
542     int i, npos=0;
543     for (i=0;i<n;i++)
544         if (tab[i]>0)
545             npos++;
546     return npos;
547 }
548
549 //fonction renvoyant l'indice du maximum du tableau tab de taille n
550 int ImaxTab(int tab[],int n)
551 {
552     int i,imax;
553     imax=0;
554     for (i=0;i<n;i++) {
555         if (tab[i]>tab[imax]) {
556             imax=i;
557         }
558     }
559     return imax;
560 }
561
562 //fonction indiquant si la valeur x est presente
563 //dans le tableau tab de taille n
564 int InTab(int tab[],int n,int x)
565 {
566     int i;
567     for (i=0;i<n;i++) {
568         if (tab[i]==x) {
569             return 1;
570         }
571     }
572     return 0; //on a parcouru tout le tableau et
573 } //on n'a jamais trouve la valeur cherchee
574
575
576
577
578

```



```

579 //fonction renvoyant la mediane du tableau T de taille n
580 int MedTab(int T[],int n)
581 {
582     int i,med,ninf;
583
584     //on initialise med au minimum du tableau
585     med=T[0];
586     for (i=1;i<n;i++)
587         if (T[i]<med)
588             med=T[i];
589
590     while(1) {
591         //on compte le nombre de valeurs inferieures a med
592         ninf=0;
593         for (i=0;i<n;i++)
594             {
595                 if (T[i]<=med)
596                     ninf++;
597             }
598         if (ninf>(n/2))
599             break; //on sort de la boucle car on a trouve la mediane
600         else
601             med++; //on incremente med
602     }
603     return med;
604 }
605
606 //fonction supprimant la valeur v du tableau T de taille n
607 //et tassant les elements restants
608 int SuppValTab(int T[], int n, int v)
609 {
610     int i,j;
611     for (i=0;i<n;i++) //on parcourt le tableau
612     {
613         if (T[i]==v) //on cherche la valeur a supprimer
614             {
615                 for (j=i;j<n-1;j++) //on remonte toutes les cases
616                     T[j]=T[j+1]; //sous la case a supprimer
617                 n--; //on decremente la taille du tableau
618                 i--; //on decremente i pour retester la case
619             }
620     }
621     return n; //on renvoie la nouvelle taille du tableau
622 } //une fois toutes les cases supprimees
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651

```

```

652 //-----
653 // TP 5 Exercice 9
654 #include<stdio.h>
655 #include <string.h>
656
657 #define NMAX 15
658
659 int main (void) {
660     int i,n, npoint,ipoint,signe, isnombre;
661     char nombre[NMAX];
662     double x;
663
664     printf("entrez un nombre : ");
665     gets(nombre);
666
667     //Initialisation des differentes variables
668     n=strlen(nombre); //longueur de la chaine
669     npoint = 0; //nombre de '.' ou ','
670     isnombre = 1; //a priori, il s'agit bien d'un nombre
671     signe = 1; //a priori, x sera positif
672     x = 0.; //valeur initiale du nombre
673
674     //Le 1er caractere doit etre un chiffre
675     //ou '+' ou '-' ou '.' ou ','
676     if (nombre[0]>='0' && nombre[0]<='9')
677         x=nombre[0]-'0';
678     else if (nombre[0]=='+')
679         signe = 1;
680     else if (nombre[0]=='-')
681         signe = -1;
682     else if (nombre[0]=='.' || nombre[0]==',') {
683         ipoint=0; //position du point ou virgule
684         npoint++;
685     }
686     else
687         isnombre = 0;
688
689     //On teste tous les autres caracteres
690     for (i=1;i<n && isnombre == 1;i++) {
691         //chaque caractere doit etre un chiffre ou '.' ou ','
692         if (nombre[i]>='0' && nombre[i]<='9') {
693             x*=10;
694             x+=nombre[i]-'0';
695         }
696         else if (nombre[i]==',' || nombre[i]=='.') {
697             if (npoint==0) { //il ne doit y avoir que un
698                 npoint++; //seul point ou virgule
699                 ipoint=i; //position du point ou virgule
700             }
701             else
702                 isnombre = 0 ;
703         }
704         else
705             isnombre = 0;
706     }
707
708     //Si la chaine entree est un nombre,
709     //on le calcule et on l'affiche
710     if (isnombre == 1) {
711         if (npoint == 1) //on a trouve une virgule
712             for (i=n-1;i>ipoint;i--) // on divise x par 10 autant de fois
713                 x/=10; //qu'il y a de chiffres apres la virgule
714
715         x*=signe; //on change le signe de x si ncessaire
716
717         printf("le nombre rentre vaut : %le\n",x);
718     }
719     else {
720         printf("ceci n'est pas un nombre valide\n");
721     }
722
723     return 0;
724 }

```