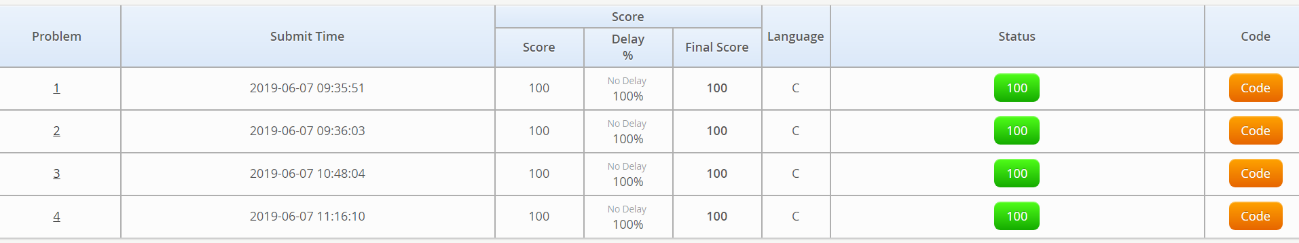
採点システムのスクリーンショット



ソースコード

演習課題１

|  |
| --- |
| #include<stdio.h>  #define NUM 1000  void bubble(int x[], int num);  void swap(int \*a, int \*b);  int n\_comp=0, n\_exchange=0; //比較回数と交換回数（グローバル変数）  int main()  {  int i,num,x[NUM];  scanf("%d",&num); //読み込む数値の数を入力  for(i=0;i<num;i++){  scanf("%d",&x[i]);  }  printf("整列前データ：");  for(i=0;i<num;i++){  printf("%4d",x[i]);  }  printf("\n");  bubble(x,num); //バブルソート  printf("整列後データ：");  for(i=0;i<num;i++){  printf("%4d",x[i]);  }  printf("\n");  printf("比較回数：%d\n",n\_comp);  printf("交換回数：%d\n",n\_exchange);  return 0;  }  void bubble(int x[],int num)  {  int i,j;  for(i=num-1;i>=0;i--){  for(j=0;j<i;j++){  n\_comp++;  if(x[j]>x[j+1]){  n\_exchange++;  swap(&x[j],&x[j+1]);  }  }  }  }  void swap(int \*a, int \*b)  {  int tmp=\*a;  \*a=\*b;  \*b=tmp;  } |

演習課題2

|  |
| --- |
| #include<stdio.h>  #define NUM 1000  void insertion(int x[], int num);  int n\_comp=0, n\_insert=0, n\_shift=0; //比較回数とシフト回数（グローバル変数）  int main()  {  int i,num,x[NUM];  scanf("%d",&num); //読み込む数値の数を入力  for(i=0;i<num;i++){  scanf("%d",&x[i]);  }  printf("整列前データ：");  for(i=0;i<num;i++){  printf("%4d",x[i]);  }  printf("\n");  insertion(x,num);  printf("整列後データ：");  for(i=0;i<num;i++){  printf("%4d",x[i]);  }  printf("\n");  printf("比較回数：%d\n",n\_comp);  printf("シフト回数：%d\n",n\_shift);  printf("挿入回数：%d\n",n\_insert);  return 0;  }  void insertion(int x[], int num) //挿入法  {  int i,j,tmp;  for(i=1;i<num;i++){  tmp=x[i];  for(j=i-1;j>=0;j--){  n\_comp++;  if(x[j]>tmp){  n\_shift++;  x[j+1]=x[j];  }else{  break;  }  }  n\_insert++;  x[j+1]=tmp;  }  } |

演習課題３

|  |
| --- |
| #include<stdio.h>  #include<stdlib.h>  #define NUM 100  void bubble(int x[], int num);  void insertion(int x[], int num);  void swap(int \*a, int \*b);  int n\_exchange=0,n\_shift=0,n\_insert=0,n\_bub\_comp=0,n\_ins\_comp=0;  int main()  {  int data\_bub[NUM],data\_ins[NUM];  int i,j,num;  for(i=0;i<5;i++){  num=(i\*20)+20;  for(j=0;j<NUM;j++){  data\_bub[j]=rand();  data\_ins[j]=data\_bub[j];  }  bubble(data\_bub,num);  insertion(data\_ins,num);  printf("データの個数 %d個\n",num);  printf("単純交換法：比較回数 %d 交換回数 %d\n",n\_bub\_comp,n\_exchange);  printf("単純挿入法：比較回数 %d シフト回数 %d 挿入回数 %d\n",n\_ins\_comp,n\_shift,n\_insert);  /\* n\_exchange=0;  n\_bub\_comp=0;  n\_ins\_comp=0;  n\_shift=0;  n\_insert=0;\*/  }  return 0;  }  void bubble(int x[], int num)  {  int i,j;  for(i=num-1;i>=0;i--){  for(j=0;j<i;j++){  n\_bub\_comp++;  if(x[i]>x[j+1]){  n\_exchange++;  swap(&x[j],&x[j+1]);  }  }  }  }  void insertion(int x[], int num)  {  int i,j,tmp;  for(i=1;i<num;i++){  tmp=x[i];  for(j=i-1;j>=0;j--){  n\_ins\_comp++;  if(x[j]>tmp){  n\_shift++;  x[j+1]=x[j];  }else{  break;  }  }  n\_insert++;  x[j+1]=tmp;  }  }  void swap(int \*a, int \*b)  {  int tmp;  tmp=\*a;  \*a=\*b;  \*b=tmp;  } |

演習課題４

|  |
| --- |
| #include<stdio.h>  #include<string.h>  typedef struct{  char name[30];  double height;  double weight;  }STUDENT;  void insertion(STUDENT gakusei[],int num);  int main()  {  STUDENT gakusei[]={  {"阿部",178.5,72.4},  {"佐藤",168.8,55.5},  {"鈴木",163.5,82.3},  {"高橋",185.0,76.5},  {"田中",175.1,64.2},  {"友近",172.3,68.2},  {"中島",179.7,77.1},  {"原田",182.1,69.3},  {"安田",174.5,58.8},  {"渡辺",176.3,65.6}  };  int i;  insertion(gakusei,10);  for(i=0;i<10;i++){  printf("%s：%f %f\n",gakusei[i].name,gakusei[i].height,gakusei[i].weight);  }  return 0;  }  void insertion(STUDENT gakusei[],int num)  {  STUDENT tmp;  int i,j;  for(i=1;i<num;i++){  strcpy(tmp.name,gakusei[i].name);  tmp.height=gakusei[i].height;  tmp.weight=gakusei[i].weight;  for(j=i-1;j>=0;j--){  if(gakusei[j].height>tmp.height){  strcpy(gakusei[j+1].name,gakusei[j].name);  gakusei[j+1].height=gakusei[j].height;  gakusei[j+1].weight=gakusei[j].weight;  }else{  break;  }  }  strcpy(gakusei[j+1].name,tmp.name);  gakusei[j+1].height=tmp.height;  gakusei[j+1].weight=tmp.weight;  }  } |

実行結果

演習課題１

|  |
| --- |
| u306145@imac-557[31]: ./8-1  10  4  5  6  -4  0  10  1  -10  10  1  整列前データ： 4 5 6 -4 0 10 1 -10 10 1  整列後データ： -10 -4 0 1 1 4 5 6 10 10  比較回数：45  交換回数：22 |

演習課題２

|  |
| --- |
| u306145@imac-557[32]: ./8-2  10  4  5  6  -4  0  10  1  -10  10  1  整列前データ： 4 5 6 -4 0 10 1 -10 10 1  整列後データ： -10 -4 0 1 1 4 5 6 10 10  比較回数：29  シフト回数：22  挿入回数：9 |

演習課題３

|  |
| --- |
| u306145@imac-557[33]: ./8-3  データの個数 20個  単純交換法：比較回数 190 交換回数 113  単純挿入法：比較回数 118 シフト回数 99 挿入回数 19  データの個数 40個  単純交換法：比較回数 970 交換回数 679  単純挿入法：比較回数 540 シフト回数 488 挿入回数 58  データの個数 60個  単純交換法：比較回数 2740 交換回数 1906  単純挿入法：比較回数 1600 シフト回数 1493 挿入回数 117  データの個数 80個  単純交換法：比較回数 5900 交換回数 4238  単純挿入法：比較回数 3162 シフト回数 2980 挿入回数 196  データの個数 100個  単純交換法：比較回数 10850 交換回数 7935  単純挿入法：比較回数 5653 シフト回数 5375 挿入回数 295 |

演習課題４

|  |
| --- |
| u306145@imac-557[34]: ./8-4  鈴木：163.500000 82.300000  佐藤：168.800000 55.500000  友近：172.300000 68.200000  安田：174.500000 58.800000  田中：175.100000 64.200000  渡辺：176.300000 65.600000  阿部：178.500000 72.400000  中島：179.700000 77.100000  原田：182.100000 69.300000  高橋：185.000000 76.500000 |

考察・感想

今回の講義ではバブルソートと単純挿入法の二つのソートの仕方を学んだ。

バブルソートではswap関数を用いて順番に入れ替えてソートするものだったが、単純挿入法では、いったん別の変数に数値を格納してから、シフトさせたり、挿入したりしてソートするものだった。バブルソートは並んでいる二つの値を比べて入れ替えるものだったので、プログラムの動きをしっかり意識してプログラムを書くことができたが、単純挿入法は、値をシフトして挿入するなど処理が少し複雑で少し頭の中で意識するのが難しかった。