プログラミング2レポート課題第2回

西田直人

2018 年 11 月 27 日

1　課題 2-1

1.1　誤っている箇所

31 行目：for 文内部を {} で囲んでいない。
24 行目：配列のアドレスを変更しようとしている。要素を inArray から outArray に移したいなら要素ごとに＝で結ぶべき。

1.2　source

```
#include <stdio.h>

void copyArray(int n, int inArray[], int outArray[]);
void printArray(int n, int a[]);
void reverseArray(int n, int inArray[], int outArray[]); //の答えb

int main(void)
{
    int a[5] = { 1, 1, 2, 3, 5 };
    int b[5];

    printf("array a[]:\n");
    printArray(5, a);

    copyArray(5, a, b);

    printf("array b[]:\n");
    printArray(5, b);

    printf("reverse of a[]:\n");
    reverseArray(5, a, b);
    printArray(5, b);

    return 0;
}

void copyArray(int n, int inArray[], int outArray[])
{
  for(int i=0; i<n; i++){
    outArray[i] = inArray[i];
  }
}

void printArray(int n, int a[])
{
    int i;

    for (i=0; i<n; i++){
```

```
        printf("%d ", a[i]);
    }
    putchar('\n');
}

void reverseArray(int n, int inArray[], int outArray[]){
  int i;

  for(i=0; i<n; i++){
    outArray[n-1-i]=inArray[i];
  }
}
```

1.3 result

```
  s1811433@7C202-P048:~/prog2/02$ cc -o a2-1 a2-1.c
  s1811433@7C202-P048:~/prog2/02$ ./a2-1
  array a[]:
  1 1 2 3 5
  array b[]:
  1 1 2 3 5
  reverse of a[]:
  5 3 2 1 1
  s1811433@7C202-P048:~/prog2/02$
```

2 課題 2-2

2.1 誤っている箇所

24 行目：i=1 となってしまっているので a[0] の値が計算に含まれていない。
24 行目：for 文の中身がでくくられていない。
25 行目：a[0] のアドレスの値を 1 ずつ増やしながら sum に加算しているという内容になってしまっている。
20 行目：sum が整数型になってしまっているので、sum/n が小数点以下切り捨てになってしまっている。

2.2 source

```
#include <stdio.h>

double average(int n, int a[]);
int maxValue(int n, int a[]);
int minValue(int n, int a[]);

int main(void)
{
  int a[5] = { 7, 1, -3, 4, 5 };
  double ave;
  int max, min, diff;

    ave = average(5, a);
    max = maxValue(5, a);
```

```
    min = minValue(5, a);
    diff = max - min;

    printf("average of a[]: %.1lf\n", ave);
    printf("max value of a[]: %d\n", max);
    printf("min value of a[]: %d\n", min);
    printf("diff of max and min: %d\n", diff);

    return 0;
}

double average(int n, int a[])
{
    int i;
    double sum;

    sum = 0;

    for (i=0; i<n; i++){
      sum += *a++;
    }

    return sum / n;
}

int maxValue(int n, int a[]){
  int m=-1000, i;

  for(i=0; i<n; i++){
    if(m<a[i]){
      m=a[i];
    }
  }

  return m;
}

int minValue(int n, int a[]){
  int m=1000, i;

  for(i=0; i<n; i++){
      if(m>a[i]){
        m=a[i];
      }
  }

  return m;
}
```

2.3 result

```
  s1811433@7C202-P048:~/prog2/02/02kadai$ cc -o a2-2 a2-2.c
  s1811433@7C202-P048:~/prog2/02/02kadai$ ./a2-2
  average of a[]: 2.8
  max value of a[]: 7
```

```
  min value of a[]: -3
  diff of max and min: 10
  s1811433@7C202-P048:~/prog2/02/02kadai$
```

3 課題 2-3

3.1 source

```
#include <stdio.h>
#include <math.h>

double determinant3x3(double A[9]);
void mult3x3(double C[9], double A[9], double B[9]);
int invert3x3(double A[9], double invA[9]);
double dot3(double u[3], double v[3]);
void cross3(double u[3], double v[3], double w[3]);
void normalize3(double u[3]);
void makeRotation3x3(double u[3], double v[3], double R[9]);


int main(){
  double A[9], B[9], C[9], u[3], v[3], w[3], R[9];
  int i, q;

  for(i=0; i<9; i++){
    A[i]=0;
    B[i]=0;
    C[i]=0;
    R[i]=0;
  }
  for(i=0;i<3;i++){
    u[i]=0;
    v[i]=0;
    w[i]=0;
  }

  printf("Input a matrix> ");

  for(i=0; i<9; i++){
    scanf("%lf", (A+i));
  }

  printf("Input another matrix> ");

  for(i=0; i<9; i++){
    scanf("%lf", (B+i));
  }

  printf("Matrix A:\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n\n",
         A[0], A[3], A[6], A[1], A[4], A[7], A[2], A[5], A[8]);

   printf("Matrix B:\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n\n",
          B[0], B[3], B[6], B[1], B[4], B[7], B[2], B[5], B[8]);

  printf("determinant of A: %lf\n\n", determinant3x3(A));
```

```
  mult3x3(C, A, B);

  printf("Matrix AxB:\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n\n",
         C[0], C[3], C[6], C[1], C[4], C[7], C[2], C[5], C[8]);

  q= invert3x3(A, C);

  if(q==0){
    printf("A^-1:\n  not invertible\n\n");
  }
  else{
    printf("Invert of matrix A:\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n\n",
           C[0], C[3], C[6], C[1], C[4], C[7], C[2], C[5], C[8]);

    mult3x3(B, A, C);

    printf("A * A^-1:\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n\n",
           B[0], B[3], B[6], B[1], B[4], B[7], B[2], B[5], B[8]);
  }



  printf("input vector u>");
  for(i=0; i<3; i++){
    scanf("%lf", u+i);
  }

  printf("input vector v>");
  for(i=0; i<3; i++){
    scanf("%lf", v+i);
  }


  printf("\ndot(u, v): %lf\n", dot3(u, v));

  cross3(u, v, w);

  printf("cross(u, v): (%lf, %lf, %lf)\n", w[0], w[1], w[2]);

  normalize3(u);
  normalize3(v);

  makeRotation3x3(u, v, R);

  printf("normalized u: (%lf, %lf, %lf)\n", u[0], u[1], u[2]);
  printf("normalized v: (%lf, %lf, %lf)\n\n", v[0], v[1], v[2]);

  printf("rotation matrix R:\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n\n",
         R[0], R[3], R[6], R[1], R[4], R[7], R[2], R[5], R[8]);

  printf("inverted rotation matrix R^-1:\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n  %lf, %lf, %lf\n\n",
         R[0], R[1], R[2], R[3], R[4], R[5], R[6], R[7], R[8]);

  return 0;
}
```

```
double determinant3x3(double A[9]){
  double det;

  det=A[0]*A[4]*A[8] +A[3]*A[7]*A[2] +A[6]*A[1]*A[5] -A[6]*A[4]*A[2] -A[0]*A[7]*A[5] -A[3]*A[1]*A[8];

  return det;
}

void mult3x3(double C[9], double A[9], double B[9]){
  C[0]= A[0]*B[0] +A[3]*B[1] +A[6]*B[2];
  C[1]= A[1]*B[0] +A[4]*B[1] +A[7]*B[2];
  C[2]= A[2]*B[0] +A[5]*B[1] +A[8]*B[2];
  C[3]= A[0]*B[3] +A[3]*B[4] +A[6]*B[5];
  C[4]= A[1]*B[3] +A[4]*B[4] +A[7]*B[5];
  C[5]= A[2]*B[3] +A[5]*B[4] +A[8]*B[5];
  C[6]= A[0]*B[6] +A[3]*B[7] +A[6]*B[8];
  C[7]= A[1]*B[6] +A[4]*B[7] +A[7]*B[8];
  C[8]= A[2]*B[6] +A[5]*B[7] +A[8]*B[8];

}

int invert3x3(double A[9], double invA[9]){
  double det;

  det= determinant3x3(A);

  if(det == 0){
    return 0;
  }
  else{
    invA[0]=(A[4]*A[8] -A[7]*A[5])/det;
    invA[3]=-(A[3]*A[8] -A[6]*A[5])/det;
    invA[6]=(A[3]*A[7] -A[6]*A[4])/det;
    invA[1]=-(A[1]*A[8] -A[7]*A[2])/det;
    invA[4]=(A[0]*A[8] -A[6]*A[2])/det;
    invA[7]=-(A[0]*A[7] -A[6]*A[1])/det;
    invA[2]=(A[1]*A[5] -A[4]*A[2])/det;
    invA[5]=-(A[0]*A[5] -A[3]*A[2])/det;
    invA[8]=(A[0]*A[4] -A[3]*A[1])/det;

    return 1;
  }
}


double dot3(double u[3], double v[3]){
  double dot=0;

  for(int i=0; i<3; i++){
    dot +=u[i]*v[i];
  }
  return dot;
}

void cross3(double u[3], double v[3], double w[3]){
  w[0]=u[1]*v[2] -u[2]*v[1];
  w[1]=u[2]*v[0] -u[0]*v[2];
```

```
  w[2]=u[0]*v[1] -u[1]*v[0];
}

void normalize3(double u[3]){
  double U;

  U= sqrt(dot3(u, u));

  u[0]= u[0]/U;
  u[1]= u[1]/U;
  u[2]= u[2]/U;
}

void makeRotation3x3(double u[3], double v[3], double R[9]){
  double e0[3], e1[3], e2[3], M;
  int i;

  for(i=0; i<3; i++){
    e0[i]=0;
    e1[i]=0;
    e2[i]=0;
  }

  normalize3(u);
  e0[0]=u[0];
  e0[1]=u[1];
  e0[2]=u[2];

  M=sqrt(dot3(v, v) +dot3(e0, e0)*dot3(v, e0)*dot3(v, e0) -2*dot3(v, e0)*dot3(v, e0));

  for(i=0; i<3; i++){
    e1[i]=(v[i] -(dot3(v, e0)*e0[i]))/M;
  }

  cross3(e0, e1, e2);

  R[0]=e0[0];
  R[1]=e0[1];
  R[2]=e0[2];
  R[3]=e1[0];
  R[4]=e1[1];
  R[5]=e1[2];
  R[6]=e2[0];
  R[7]=e2[1];
  R[8]=e2[2];
}
```

3.2 result

```
  s1811433@LI1RR-P003:~/prog2/02/02kadai$ cc -o a2-3 a2-3.c -lm
  s1811433@LI1RR-P003:~/prog2/02/02kadai$ ./a2-3
  Input a matrix> -1 4 5 3 0 -2 2 2 1
  Input another matrix> 1 2 3 4 5 6 7 8 9
  Matrix A:
  -1.000000, 3.000000, 2.000000
```

```
4.000000, 0.000000, 2.000000
5.000000, -2.000000, 1.000000

Matrix B:
1.000000, 4.000000, 7.000000
2.000000, 5.000000, 8.000000
3.000000, 6.000000, 9.000000

determinant of A: -2.000000

Matrix AxB:
11.000000, 23.000000, 35.000000
10.000000, 28.000000, 46.000000
4.000000, 16.000000, 28.000000

Invert of matrix A:
-2.000000, 3.500000, -3.000000
-3.000000, 5.500000, -5.000000
4.000000, -6.500000, 6.000000

A * A^-1:
1.000000, 0.000000, 0.000000
0.000000, 1.000000, 0.000000
0.000000, 0.000000, 1.000000

input vector u>1 2 3
input vector v>2 3 4

dot(u, v): 20.000000
cross(u, v): (-1.000000, 2.000000, -1.000000)
normalized u: (0.267261, 0.534522, 0.801784)
normalized v: (0.371391, 0.557086, 0.742781)

rotation matrix R:
0.267261, 0.872872, -0.408248
0.534522, 0.218218, 0.816497
0.801784, -0.436436, ¥-0.408248n
inverted rotation matrix R^-1:
0.267261, 0.534522, 0.801784
0.872872, 0.218218, -0.436436
-0.408248, 0.816497, -0.408248

s1811433@LI1RR-P003:~/prog2/02/02kadai$ ./a2-3
Input a matrix> 1 2 3 4 5 6 7 8 9
Input another matrix> 1 0 0 0 1 0 0 0 1
Matrix A:
1.000000, 4.000000, 7.000000
2.000000, 5.000000, 8.000000
3.000000, 6.000000, 9.000000

Matrix B:
1.000000, 0.000000, 0.000000
0.000000, 1.000000, 0.000000
0.000000, 0.000000, 1.000000

determinant of A: 0.000000
```

```
Matrix AxB:
1.000000, 4.000000, 7.000000
2.000000, 5.000000, 8.000000
3.000000, 6.000000, 9.000000

A^-1:
not invertible

input vector u>-1 2 2
input vector v>0 -1 1

dot(u, v): 0.000000
cross(u, v): (4.000000, 1.000000, 1.000000)
normalized u: (-0.333333, 0.666667, 0.666667)
normalized v: (0.000000, -0.707107, 0.707107)

rotation matrix R:
-0.333333, 0.000000, 0.942809
0.666667, -0.707107, 0.235702
0.666667, 0.707107, 0.235702
inverted rotation matrix R^-1:
-0.333333, 0.666667, 0.666667
0.000000, -0.707107, 0.707107
0.942809, 0.235702, 0.235702

s1811433@LI1RR-P003:~/prog2/02/02kadai$
```