02635 Fall 2016 — Module 2 (solutions)

Homework

- Read chapters 3 and 4 in "Beginning C"
- Read chapters 1 and 2 in "Writing Scientific Software"

Exercises — Part I

1. Do exercises 3-1, and 4-1 in "Beginning C"

Exercise 3-1

```
#include <stdio.h>

int main(void) {

  int choice=0;
  float temperature=0.0;

  printf("Temperature conversion"
         "- please select one of the following options:\n");
  printf("  1. Convert from degrees Celcius to degrees Fahrenheit\n");
  printf("  2. Convert from degrees Fahrenheit to degrees Celcius\n");
  printf("\nEnter your choice [1 or 2]: ");
  scanf("%i",&choice);

  if (choice == 1) {
    printf("Please enter temperature in degrees Celcius: ");
    scanf("%f",&temperature);
    printf("Temperature in degrees Fahrenheit: %.1f F\n", temperature*1.8+32);
  }
  else if (choice == 2) {
    printf("Please enter temperature in degrees Fahrenheit: ");
    scanf("%f",&temperature);
    printf("Temperature in degrees Celcius: %.1f C\n", (temperature-32)/1.8);
  }
  else {
    printf("Invalid choice.\n");
    return -1;
  }

  return 0;
}
```

Exercise 4-1

```c
#include <stdio.h>

int main(void) {

  int size, i, j;

  printf("Multiplication table - please enter size: ");
  scanf("%d",&size);

  // print first line with integers
  printf("      | ");
  for (j=1;j<=size;j++)
    printf("%5d ",j);
  printf("\n-------");
  for (j=1;j<=size;j++)
    printf("------");
  printf("\n");

  // print table
  for (i=1;i<=size;i++) {
    printf("%5d | ",i);
    for (j=1;j<=size;j++) {
      printf("%5d ",i*j);
    }
    printf("\n");
  }

  return 0;
}
```

Example output

```
Multiplication table - please enter size: 8
      |     1     2     3     4     5     6     7     8
-------------------------------------------------------
    1 |     1     2     3     4     5     6     7     8
    2 |     2     4     6     8    10    12    14    16
    3 |     3     6     9    12    15    18    21    24
    4 |     4     8    12    16    20    24    28    32
    5 |     5    10    15    20    25    30    35    40
    6 |     6    12    18    24    30    36    42    48
    7 |     7    14    21    28    35    42    49    56
    8 |     8    16    24    32    40    48    56    64
```

2. Do exercises 2, 3, and 4 (p. 39) in "Writing Scientific Software"

Exercise 2

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

int main(void) {

  int k;
  double x;

  for (k=0;k<=16;k++) {
    x = pow(10,-k);
    printf("f(10^(%-3d)) = %.10le\n", -k, (1-cos(x))/(x*x));
  }

  return 0;
}
```

Output

```
f(10^(0  )) = 4.5969769413e-01
f(10^(-1 )) = 4.9958347220e-01
f(10^(-2 )) = 4.9999583335e-01
f(10^(-3 )) = 4.9999995833e-01
f(10^(-4 )) = 4.9999999696e-01
f(10^(-5 )) = 5.0000004137e-01
f(10^(-6 )) = 5.0004445029e-01
f(10^(-7 )) = 4.9960036108e-01
f(10^(-8 )) = 0.0000000000e+00
f(10^(-9 )) = 0.0000000000e+00
f(10^(-10)) = 0.0000000000e+00
f(10^(-11)) = 0.0000000000e+00
f(10^(-12)) = 0.0000000000e+00
f(10^(-13)) = 0.0000000000e+00
f(10^(-14)) = 0.0000000000e+00
f(10^(-15)) = 0.0000000000e+00
f(10^(-16)) = 0.0000000000e+00
```

The cosine function can be represented by the following Taylor series

$$\cos(x) = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k}}{(2k)!}.$$

If x is close to zero, the fourth-order approximation

$$\cos(x) \approx 1 - \frac{1}{2}x^2 + \frac{1}{24}x^4$$

is reasonably accurate. Thus,

$$f(x) = \frac{1 - \cos(x)}{x^2} \approx \frac{1}{2} - \frac{1}{24}x^2$$

for x close to zero.

The round-off error in the numerator for $x = 10^{-6}$ is approximately

$$\mathbf{fl}(1 - \mathbf{fl}(\cos(x))) \approx (\mathbf{fl}(f(x)) - f(x)) \cdot x^2 \approx (0.500044 - 0.5) \cdot x^2 = 4.4 \cdot 10^{-17}.$$

Exercise 3

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

int main(void) {

  double x,y;

  printf("Input x: ");
  scanf("%lf",&x);
  printf("Input y: ");
  scanf("%lf",&y);

  if (x >= y && x > 0) {
    y /= x;
    printf("sqrt(x^2 + y^2) = %le\n", fabs(x)*sqrt(1+y*y));
  }
  else if (y > x && y > 0) {
    x /= y;
    printf("sqrt(x^2 + y^2) = %le\n", fabs(y)*sqrt(1+x*x));
  }
  else
    printf("sqrt(x^2 + y^2) = %le\n", sqrt(x*x + y*y));

  return 0;
}
```

Exercise 4

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

int main(void) {

  double a,b,c,xm,xp,det;

  printf("Solve quadratic equation a*x^2 + b*x + c == 0\n\n");

  // prompt user to enter a,b,c
  printf("Input a: ");
  scanf("%lf",&a);
  printf("Input b: ");
  scanf("%lf",&b);
  printf("Input c: ");
  scanf("%lf",&c);

  det = b*b - 4*a*c;

  if (a == 0) {
    if (b != 0)
      printf("x = %.4le\n",-c/b);
    else
      printf("a and b are both zero.\n");
    return 0;
  }
  else { // a is nonzero
    if (det < 0) { // complex roots
      printf("Complex roots\n");
      printf("x1 = %.4le + i*%.4le, ",-b/(2*a),sqrt(-det)/(2*a));
      printf("x2 = %.4le - i*%.4le\n",-b/(2*a),sqrt(-det)/(2*a));
      return 0;
    }
    else if (b*b > 10*a*c && b > 0) { // special case 1
      xm = -b - sqrt(det);
      xm /= 2*a;
      xp = c/(a*xm);
    }
    else if (b*b > 10*a*c && b < 0) { // special case 2
      xp = -b + sqrt(det);
      xp /= 2*a;
      xm = c/(a*xp);
    }
    else { // default case
      xp = (-b + sqrt(det))/(2*a);
      xm = (-b - sqrt(det))/(2*a);
    }
    printf("Real roots\n");
    printf("x1 = %.4le, x2 = %.4le\n",xp,xm);
    return 0;
  }
}
```

Exercises — Part II

Numerical integration

Exercise 1

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

#define RECTANGLE 1
#define TRAPEZOIDAL 2

int main(void) {

  double a, b, h, val = 0, x;
  int n, method;

  // Print welcome message
  printf("This program computes an approximation of the definite integral\n\n"
         "  int_a^b exp(-x^2) dx\n\n"
         "using numerical integration.\n\n");

  // Prompt user to enter integration limits
  printf("Please enter the integration limit a: ");
  scanf("%lf", &a);
  printf("Please enter the integration limit b: ");
  scanf("%lf", &b);
  // Check that a < b
  if (a>=b) {
    printf("error: a must be less than b\n");
    return -1;
  }

  // Prompt user to enter number of subintervals
  printf("Please enter the number of subintervals: ");
  scanf("%d",&n);
  // Check that n is positive
  if (n <= 0) {
    printf("error: n must be positive\n");
    return -1;
  }

  // Prompt user to choose method
  printf("Please select integration rule"
         "(%i. rectangle rule, %i. trapezoidal rule): ",
         RECTANGLE, TRAPEZOIDAL);
  scanf("%d",&method);
  // Check user input
  if (!((method == RECTANGLE) || (method == TRAPEZOIDAL))) {
    printf("error: unknown method\n");
    return -1;
  }

  // Compute approximation to definite integral and print result
  h = (b-a)/n;
  if (method == RECTANGLE) {
    for (int i=0; i<n; i++) {
      x = a + (i+0.5)*h;
      val += h*exp(-x*x);
    }
  }
  else if (method == TRAPEZOIDAL) {
    val = 0.5*h*(exp(-a*a) + exp(-b*b));
    for (int i=1; i<n-1; i++) {
      x = a+i*h;
      val += h*exp(-x*x);
    }
  }
  printf("Approximate value of definite integral: %.8le\n", val);

  return 0;
}
```

Exercise 2

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

int main(void) {

  double a, b, h, val1, val2, x;
  int n;

  // Print welcome message
  printf("This program computes an approximation of the definite integral\n\n"
         "  int_a^b exp(-x^2) dx\n\n"
         "using numerical integration.\n\n");

  // Prompt user to enter integration limits
  printf("Please enter the integration limit a: ");
  scanf("%lf", &a);
  printf("Please enter the integration limit b: ");
  scanf("%lf", &b);
  // Check that a < b
  if (a>=b) {
    printf("error: a must be less than b\n");
    return -1;
  }

  // Prompt user to enter number of subintervals
  printf("Please enter the number of subintervals: ");
  scanf("%d",&n);
  // Check that n is positive
  if (n <= 0) {
    printf("error: n must be positive\n");
    return -1;
  }

  // Compute results and print table
  printf("Parameters:\n\n  a = %.3le\n  b = %.3le\n  n = %i\n\n",a,b,n);
  printf("Results:\n\n");
  printf("%3s  %-14s  %-14s\n","n","Rectangle","Trapezoidal");
  printf("-----------------------------------\n");
  for (int i=1;i<=n;i++) {
    h = (b-a)/i;

    // Rectangle rule
    val1 = 0.0;
    for (int j=0; j<i; j++) {
      x = a + (j+0.5)*h;
      val1 += h*exp(-x*x);
    }

    // Trapezoidal rule
    val2 = 0.5*h*(exp(-a*a) + exp(-b*b));
    for (int j=1; j<i; j++) {
      x = a+j*h;
      val2 += h*exp(-x*x);
    }

    // Print row
    printf("%3i  %.8le  %.8le\n",i,val1,val2);
  }

  return 0;
}
```

Optional exercise: Monto Carlo integration

```c
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <time.h>

int main(void) {

  double a, b, val, u;
  int n;

  // Initialize random number generator
  srand(time(NULL));

  // Print welcome message
  printf("This program computes an approximation of the definite integral\n\n"
         "  int_a^b exp(-x^2) dx\n\n"
         "using Monte Carlo integration.\n\n");

  // Prompt user to enter integration limits
  printf("Please enter the integration limit a: ");
  scanf("%lf", &a);
  printf("Please enter the integration limit b: ");
  scanf("%lf", &b);
  // Check that a < b
  if (a>=b) {
    printf("error: a must be less than b\n");
    return -1;
  }

  // Prompt user to enter number of samples
  printf("Please enter the number of samples: ");
  scanf("%d",&n);
  // Check that n is positive
  if (n <= 0) {
    printf("error: n must be positive\n");
    return -1;
  }

  // Compute result
  val = 0.0;
  for (int i=1;i<=n;i++) {
    u = a + (b-a)*rand()/RAND_MAX;
    val = (1.0-1.0/i)*val + exp(-u*u)/i;
  }
  printf("Approximate value of definite integral: %.8le\n", val*(b-a));

  return 0;
}
```