Chapter 2

PE 2-1

```
/* Programming Exercise 2-1  */
#include <stdio.h>

int main(void)
{
    printf("Anton Bruckner\n");
    printf("Anton\nBruckner\n");
    printf("Anton ");
    printf("Bruckner\n");

    return 0;
}
```

PE 2-3

```
/* Programming Exercise 2-3  */
#include <stdio.h>

int main(void)
{
    int ageyears;    /* age in years */
    int agedays;     /* age in days  */
                /* large ages may require the long type */
    ageyears = 44;
    agedays = 365 * ageyears;
    printf("An age of %d years is %d days.\n", ageyears, agedays);

    return 0;
}
```

PE 2-4

```
/* Programming Exercise 2-4  */
#include <stdio.h>

void jolly(void);
void deny(void);
```

```
int main(void)
{
    jolly();
    jolly();
    jolly();
    deny();

    return 0;
}

void jolly(void)
{
    printf("For he's a jolly good fellow!\n");
}

void deny(void)
{
    printf("Which nobody can deny!\n");
}
```

PE 2-5

```
/* Programming Exercise 2-5  */
#include <stdio.h>

int main(void)
{
    int toes;

    toes = 10;

    printf("toes = %d\n", toes);
    printf("Twice toes = %d\n", 2 * toes);
    printf("toes squared = %d\n", toes * toes);

    return 0;
}

/* or create two more variables, set them to 2 * toes and toes * toes
*/
```

PE 2-7

```
/* Programming Exercise 2-7 */
#include <stdio.h>

void one_three(void);
void two(void);
int main(void)
{
    printf("starting now:\n");
    one_three();
    printf("done!\n");
    return 0;
}

void one_three(void)
{
    printf("one\n");
    two();
    printf("three\n");
}

void two(void)
{
    printf("two\n");
}
```

Chapter 3

PE 3-2

```
/* Programming Exercise 3-2  */
#include <stdio.h>

int main(void)
{
    int ascii;

    printf("Enter an ASCII code: ");
    scanf("%d", &ascii);
    printf("%d is the ASCII code for %c.\n", ascii, ascii);

    return 0;
}
```

PE 3-4

```
/* Programming Exercise 3-4  */
#include <stdio.h>

int main(void)
{
    float num;
    printf("Enter a floating-point value: ");
    scanf("%f", &num);
    printf("fixed-point notation: %f\n", num);
    printf("exponential notation: %e\n", num);

    return 0;
}
```

PE 3-6

```
/* Programming Exercise 3-6  */
#include <stdio.h>

int main(void)
{
    float mass_mol = 3.0e-23;   /* mass of water molecule in grams */
    float mass_qt = 950;        /* mass of quart of water in grams */
    float quarts;
    float molecules;

    printf("Enter the number of quarts of water: ");
    scanf("%f", &quarts);
    molecules = quarts * mass_qt / mass_mol;
    printf("%f quarts of water contain %e molecules.\n", quarts,
            molecules);

    return 0;
}
```

Chapter 4

PE 4-1

```
/* Programming Exercise 4-3 */
#include <stdio.h>

int main(void)
{
    char fname[40];
    char lname[40];

    printf("Enter your first name: ");
    scanf("%s", fname);
    printf("Enter your last name: ");
    scanf("%s", lname);
    printf("%s, %s\n", lname, fname);

    return 0;
}
```

PE 4-4

```
/* Programming Exercise 4-4 */
#include <stdio.h>

int main(void)
{
    float height;
    char name[40];

    printf("Enter your height in inches: ");
    scanf("%f", &height);
    printf("Enter your name: ");
    scanf("%s", name);
    printf("%s, you are %.3f feet tall\n", name, height / 12.0);

    return 0;
}
```

PE 4-6

```
/* Programming Exercise 4-6 */
#include <stdio.h>
#include <float.h>
```

```
int main(void)
{
    float ot_f = 1.0 / 3.0;
    double ot_d = 1.0 / 3.0;

    printf(" float values: ");
    printf("%.4f %.12f %.16f\n", ot_f, ot_f, ot_f);
    printf("double values: ");
    printf("%.4f %.12f %.16f\n", ot_d, ot_d, ot_d);
    printf("FLT_DIG: %d\n", FLT_DIG);
    printf("DBL_DIG: %d\n", DBL_DIG);
    return 0;
}
```

Chapter 5

PE 5-1

```
/* Programming Exercise 5-1 */
#include <stdio.h>

int main(void)
{
    const int minperhour = 60;
    int minutes, hours, mins;

    printf("Enter the number of minutes to convert: ");
    scanf("%d", &minutes);
    while (minutes > 0 )
    {
        hours = minutes / minperhour;
        mins = minutes % minperhour;
        printf("%d minutes = %d hours, %d minutes\n", minutes, hours,
mins);
        printf("Enter next minutes value (0 to quit): ");
        scanf("%d", &minutes);
    }
    printf("Bye\n");

    return 0;
}
```

PE 5-3

```
/* Programming Exercise 5-3 */
#include <stdio.h>

int main(void)
{
    const int daysperweek = 7;
    int days, weeks, day_rem;

    printf("Enter the number of days: ");
    scanf("%d", &days);
    weeks = days / daysperweek;
    day_rem = days % daysperweek;

    printf("%d days are %d weeks and %d days.\n", days, weeks, day_rem);

    return 0;
}
```

PE 5-5

```
/* Programming Exercise 5-5 */
#include <stdio.h>
int main(void)                  /* finds sum of first n integers */
{
  int count, sum;
  int n;

  printf("Enter the upper limit: ");
  scanf("%d", &n);
  count = 0;
  sum = 0;
  while (count++ < n)
     sum = sum + count;
  printf("sum = %d\n", sum);
  return 0;
}
```

PE 5-7

```
/* Programming Exercise 5-7 */
```

```
#include <stdio.h>
void showCube(double x);
int main(void)     /* finds cube of entered number */
{
     double val;

     printf("Enter a floating-point value: ");
     scanf("%lf", &val);
     showCube(val);

    return 0;
}

void showCube(double x)
{
    printf("The cube of %e is %e.\n", x, x*x*x );
}
```

Chapter 6

PE 6-1

```
/* pe6-1.c */
/* this implementation assumes the character codes */
/* are sequential, as they are in ASCII.           */
#include <stdio.h>
#define SIZE 26
int main( void )
{
    char lcase[SIZE];
    int i;

    for (i = 0; i < SIZE; i++)
        lcase[i] = 'a' + i;
    for (i = 0; i < SIZE; i++)
        printf("%c", lcase[i]);
    printf("\n");

    return 0;
}
```

PE 6-3

```
/* pe6-3.c */
/* this implementation assumes the character codes */
/* are sequential, as they are in ASCII.           */

#include <stdio.h>

int main( void )
{
    char let = 'F';
    char start;
    char end;

    for (end = let; end >= 'A'; end--)
    {
        for (start = let; start >= end; start--)
            printf("%c", start);
        printf("\n");
    }

    return 0;
}
```

PE 6-5

```
/* pe6-5.c */

#include <stdio.h>

int main( void )
{
    int lower, upper, index;
    int square, cube;

    printf("Enter starting integer: ");
    scanf("%d", &lower);
    printf("Enter ending integer: ");
    scanf("%d", &upper);

    printf("%5s %10s %15s\n", "num", "square", "cube");
    for (index = lower; index <= upper; index++)
    {
        square = index * index;
        cube = index * square;
```

```
        printf("%5d %10d %15d\n", index, square, cube);
    }

    return 0;
}
```

PE 6-7

```
/* pe6-7.c */

#include <stdio.h>

int main( void )
{
    double n, m;
    double res;

    printf("Enter a pair of numbers: ");

    while (scanf("%lf %lf", &n, &m) == 2)
    {
        res = (n - m) / (n * m);
        printf("(%.3g - %.3g)/(%.3g*%.3g) = %.5g\n", n, m, n, m, res);
        printf("Enter next pair (non-numeric to quit): ");
    }

    return 0;
}
```

PE 6-10

```
/* pe6-10.c */

#include <stdio.h>
#define SIZE 8
int main( void )
{
    int vals[SIZE];
    int i;

    printf("Please enter %d integers.\n", SIZE);
    for (i = 0; i < SIZE; i++)
```

```
        scanf("%d", &vals[i]);
    printf("Here, in reverse order, are the values you entered:\n");
    for (i = SIZE - 1; i > 0; i--)
        printf("%d ", vals[i]);
    printf("\n");

    return 0;
}
```

PE 6-12

```
/* pe6-12.c */
/* This version starts with the 0 power */

#include <stdio.h>

#define SIZE 8
int main( void )
{
    int twopows[SIZE];
    int i;
    int value = 1;      /* 2 to the 0 */

    for (i = 0; i < SIZE; i++)
    {
        twopows[i] = value;
        value *= 2;
    }

    i = 0;
    do
    {
        printf("%d ", twopows[i]);
        i++;
    } while (i < SIZE);
    printf("\n");

    return 0;
}
```

PE 6-13

```
/* pe-13.c */
/* Programming Exercise 6-13 */
#include <stdio.h>

#define SIZE 8
int main(void)
{
    double arr[SIZE];
    double arr_cumul[SIZE];
    int i;

    printf("Enter %d numbers:\n", SIZE);

    for (i = 0; i < SIZE; i++)
    {
        printf("value #%d: ", i + 1);
        scanf("%lf", &arr[i]);
    /* or scanf("%lf", arr + i);    */
    }

    arr_cumul[0] = arr[0];      /* set first element */
    for (i = 1; i < SIZE; i++)
        arr_cumul[i] = arr_cumul[i-1] + arr[i];

    for (i = 0; i < SIZE; i++)
        printf("%8g ", arr[i]);
    printf("\n");
    for (i = 0; i < SIZE; i++)
        printf("%8g ", arr_cumul[i]);
    printf("\n");


    return 0;
}


PE 6-15

/* pe6-15.c */

#include <stdio.h>

#define RATE_SIMP 0.10
#define RATE_COMP 0.05
```

```
#define INIT_AMT 100.0
int main( void )
{
    double daphne = INIT_AMT;
    double deidre = INIT_AMT;
    int years = 0;

    while (deidre <= daphne)
    {
        daphne += RATE_SIMP * INIT_AMT;
        deidre += RATE_COMP * deidre;
        ++years;
    }
    printf("Investment values after %d years:\n", years);
    printf("Daphne: $%.2f\n", daphne);
    printf("Deidre: $%.2f\n", deidre);

    return 0;
}
```

Chapter 7

PE 7-1

```
/* Programming Exercise 7-1 */
#include <stdio.h>

int main(void)
{
     char ch;
     int sp_ct = 0;
     int nl_ct = 0;
     int other = 0;

     while ((ch = getchar()) != '#')
     {
         if (ch == ' ')
             sp_ct++;
         else if (ch == '\n')
             nl_ct++;
         else
             other++;
     }
```

```
    printf("spaces %d, newlines: %d, others: %d\n", sp_ct, nl_ct,
other);

    return 0;
}
```

PE 7-3

```
/* Programming Exercise 7-3 */
#include <stdio.h>

int main(void)
{
    int n;
    double sumeven = 0.0;
    int ct_even = 0;
    double sumodd = 0.0;
    int ct_odd = 0;

    while (scanf("%d", &n) == 1 && n != 0)
    {
        if (n % 2 == 1)
        {
            sumodd += n;
            ++ct_odd;
        }
        else
        {
            sumeven += n;
            ++ct_even;
        }
    }
    printf("Number of evens: %d", ct_even);
    if (ct_even > 0)
        printf("  average: %g", sumeven / ct_even);
    putchar('\n');

    printf("Number of odds: %d", ct_odd);
    if (ct_odd > 0)
        printf("  average: %g", sumodd / ct_odd);
    putchar('\n');
    printf("\ndone\n");
```

```
    return 0;
}
```

PE 7-5

```
/* Programming Exercise 7-5 */
#include <stdio.h>

int main(void)
{
    char ch;
    int ct1 = 0;
    int ct2 = 0;

    while ((ch = getchar()) != '#')
    {
        switch(ch)
        {
            case '.'  : putchar('!');
                        ++ct1;
                        break;

            case '!'  : putchar('!');
                        putchar('!');
                        ++ct2;
                        break;
            default   :  putchar(ch);
        }
    }
    printf("%d replacements of . with !\n", ct1);
    printf("%d replacements of ! with !!\n", ct2);

    return 0;
}
```

PE 7-7

```
/* Programming Exercise 7-7 */
#include <stdio.h>

#define BASEPAY     10      /* $10 per hour         */
#define BASEHRS     40      /* hours at basepay     */
```

```
#define OVERTIME    1.5     /* 1.5 time                */
#define AMT1        100     /* 1st rate tier           */
#define AMT2        150     /* 2st rate tier           */
#define RATE1       0.15    /* rate for 1st tier       */
#define RATE2       0.20    /* rate for 2nd tier       */
#define RATE3       0.25    /* rate for 3rd tier       */
int main(void)
{
    double hours;
    double gross;
    double net;
    double taxes;

    printf("Enter the number of hours worked this week: ");
    scanf("%lf", &hours);
    if (hours <= BASEHRS)
        gross = hours * BASEPAY;
    else
        gross = BASEHRS * BASEPAY + (hours - BASEHRS) * BASEPAY *
OVERTIME;
    if (gross <= AMT1)
        taxes = gross * RATE1;
    else if (gross <= AMT1 + AMT2)
        taxes = AMT1 * RATE1 + (gross - AMT1) * RATE2;
    else
        taxes = AMT1 * RATE1 + AMT2 * RATE2 + (gross - AMT1 - AMT2) *
RATE3;
    net = gross - taxes;
    printf("gross: $%.2f; taxes: $%.2f; net: $%.2f\n", gross, taxes,
net);

    return 0;
}
```

PE 7-9

```
/* Programmming Exercise 7-9 */
#include <stdio.h>
#define NO 0
#define YES 1
int main(void)
{
   long num;                    /* value to be checked */
```

```
    long div;              /* potential divisors  */
    long lim;              /* limit to its values */
    int prime;

    printf("Please enter limit to values to be checked: ");
    printf("Enter q to quit.\n");
    while (scanf("%ld", &lim) == 1 && lim > 0)
    {
          for (num = 2; num <= lim; num++)
          {
           for (div = 2, prime = YES; (div * div) <= num; div++)
              if (num % div == 0)
                 prime = NO;           /* number is not prime  */
           if (prime == YES)
              printf("%ld is prime.\n", num);
       }
       printf("Please enter another limit: ");
       printf("Enter q to quit.\n");
    }
    return 0;
}
```

PE 7-11

```
/* pe7-11.c */
/* Programming Exercise 7-11 */
#include <stdio.h>
#include <ctype.h>
int main(void)
{
  const double price_artichokes = 1.25;
  const double price_beets = 0.65;
  const double price_carrots = 0.89;
  const double DISCOUNT_RATE = 0.05;

  char ch;
  double lb_artichokes;
  double lb_beets;
  double lb_carrots;
  double lb_total;

  double cost_artichokes;
  double cost_beets;
```

```
double cost_carrots;
double cost_total;
double final_total;
double discount;
double shipping;

printf("Enter a to buy artichokes, b for beets, ");
printf("c for carrots, q to quit: ");
while ((ch = getchar()) != 'q' && ch != 'Q')
{
    if (ch == '\n')
        continue;
    while (getchar() != '\n')
        continue;
     ch = tolower(ch);
    switch (ch)
    {
        case 'a' : printf("Enter pounds of artichokes: ");
                   scanf("%lf", &lb_artichokes);
                   break;
         case 'b' : printf("Enter pounds of beets: ");
                   scanf("%lf", &lb_beets);
                   break;

         case 'c' : printf("Enter pounds of carrots: ");
                   scanf("%lf", &lb_carrots);
                   break;
      default  : printf("%c is not a valid choice.\n");
  }
  printf("Enter a to buy artichokes, b for beets, ");
  printf("c for carrots, q to quit: ");
}

cost_artichokes = price_artichokes * lb_artichokes;
cost_beets = price_beets * lb_beets;
cost_carrots = price_carrots * lb_carrots;
cost_total = cost_artichokes + cost_beets + cost_carrots;
lb_total = lb_artichokes + lb_beets + lb_carrots;
if (lb_total <= 0)
    shipping = 0.0;
else if (lb_total < 5.0)
    shipping = 3.50;
else if (lb_total < 20)
    shipping = 10.0;
```

```
    else
        shipping = 8.0 + 0.1 * lb_total;
    if (cost_total > 100.0)
        discount = DISCOUNT_RATE * cost_total;
    else
      discount = 0.0;
    final_total = cost_total + shipping - discount;
    printf("Your order:\n");
    printf("%.2f lbs of artichokes at $%.2f per pound:$ %.2f\n",
              lb_artichokes, price_artichokes, cost_artichokes);
    printf("%.2f lbs of beets at $%.2f per pound: $%.2f\n",
              lb_beets, price_beets, cost_beets);
    printf("%.2f lbs of carrots at $%.2f per pound: $%.2f\n",
              lb_carrots, price_carrots, cost_carrots);
    printf("Total cost of vegetables: $%.2f\n", cost_total);
    if (cost_total > 100)
        printf("Volume discount: $%.2f\n", discount);
    printf("Shipping: $%.2f\n", shipping);
    printf("Total charges: $%.2f\n", final_total);

    return 0;
}
```

Chapter 8

PE 8-1

```
/* Programming Exercise 8-1 */

#include <stdio.h>
int main(void)
{
    int ch;
    int ct = 0;

    while ((ch = getchar()) != EOF)
        ct++;
    printf("%d characters read\n", ct);

    return 0;
}
```

PE 8-3

```
/* Programming Exercise 8-3 */
/* Using ctype.h eliminates need to assume ASCII coding */
#include <stdio.h>
#include <ctype.h>
int main(void)
{
    int ch;
    int uct = 0;
    int lct = 0;

    while ((ch = getchar()) != EOF)
        if (isupper(ch))
            uct++;
        else if (islower(ch))
            lct++;
    printf("%d uppercase characters read\n", uct);
    printf("%d lowercase characters read\n", lct);

    return 0;
}

/*
 or you could use
 if (ch >= 'A' && ch <= 'Z')
     uct++;
 else if (ch >= 'a' && ch <= 'z')
     lct++;
*/
```

PE 8-5

```
/* Programming Exercise 8-5 */
/* binaryguess.c -- an improved number-guesser */
#include <stdio.h>
#include <ctype.h>
int main(void)
{
  int high = 100;
  int low = 1;
  int guess = (high + low) / 2;
  char response;
```

```
    printf("Pick an integer from 1 to 100. I will try to guess ");
    printf("it.\nRespond with a y if my guess is right, with");
    printf("\na h if it is high, and with an l if it is low.\n");
    printf("Uh...is your number %d?\n", guess);
    while ((response = getchar()) != 'y')       /* get response */
    {
        if (response == '\n')
            continue;
        if (response != 'h' && response != 'l')
        {
            printf("I don't understand that response. Please enter h
for\n");
            printf("high, l for low, or y for correct.\n");
            continue;
        }

        if (response == 'h')
            high = guess - 1;
        else if (response == 'l')
            low = guess + 1;
        guess = (high + low) / 2;
      printf("Well, then, is it %d?\n", guess);
    }
    printf("I knew I could do it!\n");
    return 0;
}
```

PE 8-7

```
/* Programming Exercise 8-7 */
#include <stdio.h>
#include <ctype.h>

#define BASEPAY1    8.75    /* $8.75 per hour       */
#define BASEPAY2    9.33    /* $9.33 per hour       */
#define BASEPAY3    10.00   /* $10.00 per hour      */
#define BASEPAY4    11.20   /* $11.20 per hour      */
#define BASEHRS     40      /* hours at basepay     */
#define OVERTIME    1.5     /* 1.5 time             */
#define AMT1        300     /* 1st rate tier        */
#define AMT2        150     /* 2st rate tier        */
#define RATE1       0.15    /* rate for 1st tier    */
```

```
#define RATE2       0.20    /* rate for 2nd tier    */
#define RATE3       0.25    /* rate for 3rd tier    */

int getfirst(void);
void menu(void);

int main(void)
{
    double hours;
    double gross;
    double net;
    double taxes;
    double pay;
    char response;


    menu();
    while ((response = getfirst()) != 'q')
    {
        if (response == '\n')           /* skip over newlines     */
            continue;
        response = tolower(response);   /* accept A as a, etc.    */
        switch (response)
        {
            case 'a'  :  pay = BASEPAY1; break;
            case 'b'  :  pay = BASEPAY2; break;
            case 'c'  :  pay = BASEPAY3; break;
            case 'd'  :  pay = BASEPAY4; break;
            default   :  printf("Please enter a, b, c, d, or q.\n");
                         menu();
                         continue;   /* go to beginning of loop */
        }
        printf("Enter the number of hours worked this week: ");
        scanf("%lf", &hours);
        if (hours <= BASEHRS)
            gross = hours * pay;
        else
            gross = BASEHRS * pay + (hours - BASEHRS) * pay * OVERTIME;
        if (gross <= AMT1)
            taxes = gross * RATE1;
        else if (gross <= AMT1 + AMT2)
            taxes = AMT1 * RATE1 + (gross - AMT1) * RATE2;
        else
```

```
            taxes = AMT1 * RATE1 + AMT2 * RATE2 + (gross - AMT1 - AMT2)
*
                    RATE3;
        net = gross - taxes;
        printf("gross: $%.2f; taxes: $%.2f; net: $%.2f\n", gross, taxes,
                net);
        menu();
    }
    printf("Done.\n");

    return 0;
}

void menu(void)
{

printf("********************************************************"
        "*********\n");
    printf("Enter the number corresponding to the desired pay rate"
        " or action:\n");
    printf("a)  $%4.2f/hr                b)  $%4.2f/hr\n", BASEPAY1,
        BASEPAY2);
    printf("c) $%5.2f/hr                 d) $%5.2f/hr\n", BASEPAY3,
        BASEPAY4);
    printf("q) quit\n");

printf("********************************************************"
        "*********\n");
}

int getfirst(void)
{
    int ch;

    ch = getchar();
    while (isspace(ch))
        ch = getchar();
    while (getchar() != '\n')
        continue;
    return ch;
}
```

Chapter 9

PE 9-1

```
/* Programming Exercise 9-1 */
#include <stdio.h>

double min(double a, double b);
int main(void)
{
    double x, y;

    printf("Enter two numbers (q to quit): ");
    while (scanf("%lf %lf", &x, &y) == 2)
    {
        printf("The smaller number is %f.\n", min(x,y));
        printf("Next two values (q to quit): ");
    }
    printf("Bye!\n");

    return 0;
}

double min(double a, double b)
{
    return a < b ? a : b;

}
/* alternative implementation
double min(double a, double b)
{
    if (a < b)
        return a;
    else
        return b;
}
*/
```

PE 9-3

```
/* Programming Exercise 9-3 */
#include <stdio.h>

void chLineRow(char ch, int c, int r);
int main(void)
```

```
{
    char ch;
    int col, row;

    printf("Enter a character (# to quit): ");
    while ( (ch = getchar()) != '#')
    {
        if (ch == '\n')
            continue;
        printf("Enter number of columns and number of rows: ");
        if (scanf("%d %d", &col, &row) != 2)
            break;
        chLineRow(ch, col, row);
        printf("\nEnter next character (# to quit): ");
    }
    printf("Bye!\n");

    return 0;
}

void chLineRow(char ch, int c, int r)
{
    int col, row;

    for (row = 0; row < r ; row++)
    {
        for (col = 0; col < c; col++)
            putchar(ch);
        putchar('\n');
    }
    return;
}


PE 9-5

/* Programming Exercise 9-5 */
#include <stdio.h>

void larger_of(double *p1, double *p2);
int main(void)
{
    double x, y;
```

```
    printf("Enter two numbers (q to quit): ");
    while (scanf("%lf %lf", &x, &y) == 2)
    {
        larger_of(&x, &y);
        printf("The modified values are %f and %f.\n", x, y);
        printf("Next two values (q to quit): ");
    }
    printf("Bye!\n");

    return 0;
}

void larger_of(double *p1, double *p2)
{
    double temp = *p1 > *p2 ? *p1 : *p2;
    *p1= *p2 = temp;
}
```

PE 9-7

```
/* Programming Exercise 9-7 */
#include <stdio.h>
double power(double a, int b);  /* ANSI prototype */
int main(void)
{
  double x, xpow;
  int n;

  printf("Enter a number and the integer power");
  printf(" to which\nthe number will be raised. Enter q");
  printf(" to quit.\n");
  while (scanf("%lf%d", &x, &n) == 2)
  {
       xpow = power(x,n);       /* function call          */
       printf("%.3g to the power %d is %.5g\n", x, n, xpow);
       printf("Enter next pair of numbers or q to quit.\n");
  }
  printf("Hope you enjoyed this power trip -- bye!\n");
  return 0;
}

double power(double a, int b)   /* function definition    */
```

```
{
  double pow = 1;
  int i;

  if (b == 0)
  {
      if (a == 0)
          printf("0 to the 0 undefined; using 1 as the value\n");
      pow = 1.0;
  }
  else if (a == 0)
      pow = 0.0;
  else if (b > 0)
      for(i = 1; i <= b; i++)
       pow *= a;
  else    /* b < 0 */
      pow = 1.0 / power(a, - b);
  return pow;                    /* return the value of pow  */
}
```

PE 9-9

```
/* Programming Exercise 9-9 */
#include <stdio.h>
void to_base_n(int x, int base);
int main(void)
{
  int number;
  int b;

  printf("Enter an integer (q to quit):\n");
  while (scanf("%d", &number) == 1)
  {
       printf("Enter number base (2-10): ");
       scanf("%d", &b);
     printf("Base %d equivalent: ", b);
     to_base_n(number, b);
     putchar('\n');
     printf("Enter an integer (q to quit):\n");
  }
  return 0;
}
```

```
void to_base_n(int x, int base)  /* recursive function */
{
  int r;

  r = x % base;
  if (x >= 2)
     to_base_n(x / base, base);
  putchar('0' + r);
  return;
}
```

Chapter 10

PE 10-1

```
/* Programming Exercise 10-1 */
#include <stdio.h>
#define MONTHS 12    /* number of months in a year */
#define YRS   5      /* number of years of data    */
int main(void)
{
 /* initializing rainfall data for 1990 - 1994 */
 const float rain[YRS][MONTHS] = {
 {10.2, 8.1, 6.8, 4.2, 2.1, 1.8, 0.2, 0.3, 1.1, 2.3, 6.1, 7.4},
 {9.2, 9.8, 4.4, 3.3, 2.2, 0.8, 0.4, 0.0, 0.6, 1.7, 4.3, 5.2},
 {6.6, 5.5, 3.8, 2.8, 1.6, 0.2, 0.0, 0.0, 0.0, 1.3, 2.6, 4.2},
 {4.3, 4.3, 4.3, 3.0, 2.0, 1.0, 0.2, 0.2, 0.4, 2.4, 3.5, 6.6},
 {8.5, 8.2, 1.2, 1.6, 2.4, 0.0, 5.2, 0.9, 0.3, 0.9, 1.4, 7.2}
 };
 int year, month;
 float subtot, total;

 printf(" YEAR    RAINFALL  (inches)\n");
 for (year = 0, total = 0; year < YRS; year++)
 {             /* for each year, sum rainfall for each month */
    for (month = 0, subtot = 0; month < MONTHS; month++)
       subtot += *(*(rain + year) + month);
    printf("%5d %15.1f\n", 1990 + year, subtot);
    total += subtot;                  /* total for all years */
 }
 printf("\nThe yearly average is %.1f inches.\n\n", total/YRS);
 printf("MONTHLY AVERAGES:\n\n");
```

```
    printf(" Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct ");
    printf(" Nov  Dec\n");

    for (month = 0; month < MONTHS; month++)
    {               /* for each month, sum rainfall over years */
        for (year = 0, subtot =0; year < YRS; year++)
            subtot += *(*(rain + year) + month);
        printf("%4.1f ", subtot/YRS);
    }
    printf("\n");
    return 0;
}
```

PE 10-3

```
/* Programming Exercise 10-3 */
#include <stdio.h>
#define LEN 10

int max_arr(const int ar[], int n);
void show_arr(const int ar[], int n);

int main(void)
{
    int orig[LEN] = {1,2,3,4,12,6,7,8,9,10};
    int max;

    show_arr(orig, LEN);
    max = max_arr(orig, LEN);
    printf("%d = largest value\n", max);

    return 0;
}

int max_arr(const int ar[], int n)
{
    int i;
    int max = ar[0];
/* don't use 0 as initial max value -- fails if all array values are neg
*/

    for (i = 1; i < n; i++)
        if (max < ar[i])
```

```
            max = ar[i];
    return max;
}

void show_arr(const int ar[], int n)
{
    int i;

    for (i = 0; i < n; i++)
        printf("%d ", ar[i]);
    putchar('\n');
}
```

PE 10-5

```
/* Programming Exercise 10-5 */
#include <stdio.h>
#define LEN 10

float max_diff(const float ar[], int n);
void show_arr(const float ar[], int n);

int main(void)
{
    float orig[LEN] = {1.1,2,3,4,12,6,7,8,9,10};
    float max;

    show_arr(orig, LEN);
    max = max_diff(orig, LEN);
    printf("%g = maximum difference\n", max);

    return 0;
}

float max_diff(const float ar[], int n)
{
    int i;
    float max = ar[0];
    float min = ar[0];

    for (i = 1; i < n; i++)
    {
```

```
        if (max < ar[i])
            max = ar[i];
        else if (min > ar[i])
            min = ar[i];
    }
    return max - min;
}

void show_arr(const float ar[], int n)
{
    int i;

    for (i = 0; i < n; i++)
        printf("%g ", ar[i]);
    putchar('\n');
}
```

PE 10-7

```
/* Programming Exercise 10-7 */
#include <stdio.h>
#define LEN1 7
#define LEN2 3

void copy_arr(int ar1[], const int ar2[], int n);
void show_arr(const int ar[], int n);

int main(void)
{
    int orig[LEN1] = {1,2,3,4,5,6,7};
    int copy[LEN2];

    show_arr(orig, LEN1);
    copy_arr(copy, orig + 2, LEN2);
    show_arr(copy, LEN2);

    return 0;
}

void copy_arr(int ar1[], const int ar2[], int n)
{
    int i;
```

```
    for (i = 0; i < n; i++)
        ar1[i] = ar2[i];
}

void show_arr(const int ar[], int n)
{
    int i;

    for (i = 0; i < n; i++)
        printf("%d ", ar[i]);
    putchar('\n');
}
```

PE 10-10

```
/* Programming Exercise 10-10 */
#include <stdio.h>
#define ROWS 3
#define COLS 5

void times2(int ar[][COLS], int r);
void showarr2(int ar[][COLS], int r);

int main(void)
{
    int stuff[ROWS][COLS] = {    {1, 2, 3, 4, 5},
                                 {6, 7, 8, 9, 10},
                                 {11, 12, 13, 14, 15}
                             };
    showarr2(stuff, ROWS);
    putchar('\n');
    times2(stuff, ROWS);
    showarr2(stuff, ROWS);

    return 0;
}

void times2(int ar[][COLS], int r)
{
    int row, col;

    for (row = 0; row < r; row++)
```

```
        for (col = 0; col < COLS; col++)
            ar[row][col] *= 2;

}

void showarr2(int ar[][COLS], int r)
{
    int row, col;

    for (row = 0; row < r; row++)
    {
        for (col = 0; col < COLS; col++)
            printf("%d ", ar[row][col]);
        putchar('\n');
    }
}
```

PE 10-13

```
/* Programming Exercise 10-13 */
#include <stdio.h>
#define ROWS 3
#define COLS 5

void store(double ar[], int n);
double average2d(int rows, int cols, double ar[rows][cols]);
double max2d(int rows, int cols, double ar[rows][cols]);
void showarr2(int rows, int cols, double ar[rows][cols]);
double average(const double ar[], int n);

int main(void)
{
    double stuff[ROWS][COLS];
    int row;

    for (row = 0; row < ROWS; row++)
    {
        printf("Enter %d numbers for row %d\n", COLS, row + 1);
        store(stuff[row], COLS);
    }

    printf("array contents:\n");
    showarr2(ROWS, COLS, stuff);
```

```
    for (row = 0; row < ROWS; row++)
        printf("average value of row %d = %g\n", row + 1,
average(stuff[row], COLS));
    printf("average value of all rows = %g\n", average2d(ROWS, COLS,
stuff));
    printf("largest value = %g\n", max2d(ROWS, COLS, stuff));
    printf("Bye!\n");
    return 0;
}

void store(double ar[], int n)
{
    int i;
    for (i = 0; i < n; i++)
    {
        printf("Enter value #%d: ", i + 1);
        scanf("%lf", & ar[i]);
    }
}


double average2d(int rows, int cols, double ar[rows][cols])
{
    int r, c;
    double sum = 0.0;

    for (r = 0; r < rows; r++)
        for (c = 0; c < cols; c++)
            sum += ar[r][c];
    if (rows * cols > 0)
        return sum / (rows * cols);
    else
        return 0.0;
}

double max2d(int rows, int cols, double ar[rows][cols])
{
    int r, c;
    double max = ar[0][0];

    for (r = 0; r < rows; r++)
        for (c = 0; c < cols; c++)
            if (max < ar[r][c])
```

```
                max = ar[r][c];
    return max;

}

void showarr2(int rows, int cols, double ar[rows][cols])
{
    int row, col;

    for (row = 0; row < rows; row++)
    {
        for (col = 0; col < cols; col++)
            printf("%g ", ar[row][col]);
        putchar('\n');
    }
}

double average(const double ar[], int n)
{
    int i;
    double sum = 0.0;

    for (i = 0; i < n; i++)
        sum += ar[i];
    if (n > 0)
        return sum / n;
    else
        return 0.0;
}
```

Chapter 11

PE 11-1

```
/* Programming Exercise 11-1 */
#include <stdio.h>
#define LEN 10
char * getnchar(char * str, int n);
int main(void)
{
    char input[LEN];
    char *chk;
```

```
    chk = getnchar(input, LEN - 1);
    if (chk == NULL)
        puts("Input failed.");
    else
        puts(input);
    puts("Done.\n");

    return 0;
}

char * getnchar(char * str, int n)
{
    int i;
    int ch;

    for (i = 0; i < n; i++)
    {
        ch = getchar();
        if (ch != EOF)
            str[i] = ch;
        else
            break;
    }
    if (ch == EOF)
        return NULL;
    else
    {
        str[i] = '\0';
        return str;
    }
}
```

PE 11-3

```
/* Programming Exercise 11-3 */
#include <stdio.h>
#define LEN 80
char * getword(char * str);
int main(void)
{
    char input[LEN];
    char *chk;
```

```
    while (getword(input) != NULL)
        puts(input);
    puts("Done.\n");

    return 0;
}

#include <ctype.h>
char * getword(char * str)
{
    int i;
    int ch;

    while ((ch = getchar()) != EOF && !isspace(ch))
        *str++ = ch;
    *str = '\0';
    if (ch == EOF)
        return NULL;
    else
    {
        while (ch != '\n')
            ch = getchar();
        return str;
    }
}
```

PE 11-5

```
/* Programming Exercise 11-5 */
#include <stdio.h>
#define LEN 80
int is_within(const char * str, char c);
int main(void)
{
    char input[LEN];
    char ch;
    int found;;

    printf("Enter a string: ");
    while (gets(input) && input[0] != '\0')
    {
        printf("Enter a character: ");
```

```
        ch = getchar();
        while (getchar() != '\n')
            continue;
        found = is_within(input, ch);
        if (found == 0)
            printf("%c not found in string.\n", ch);
        else
            printf("%c found in string %s\n", ch, input);
        printf("Next string: ");
    }
    puts("Done.\n");

    return 0;
}

int is_within(const char * str, char ch)
{
    while (*str != ch && *str != '\0')
        str++;
    return *str;    /* = 0 if \0 reached, non-zero otherwise */
}
```

PE 11-7

```
/* Programming Exercise 11-7 */
#include <stdio.h>
#define LEN 20
char * string_in(const char * s1, const char * s2);
int main(void)
{
    char orig[LEN] = "transportation";
    char * find;

    puts(orig);
    find = string_in(orig, "port");
    if (find)
        puts(find);
    else
        puts("Not found");
    find = string_in(orig, "part");
    if (find)
        puts(find);
    else
```

```
        puts("Not found");

    return 0;
}

#include <string.h>
char * string_in(const char * s1, const char * s2)
{
    int l2 = strlen(s2);
    int tries;              /* maximum number of comparisons    */
    int nomatch = 1;     /* set to 0 if match is found       */

    tries = strlen(s1) + 1 - l2;
    if (tries > 0)
        while (( nomatch = strncmp(s1, s2, l2)) && tries--)
            s1++;
    if (nomatch)
        return NULL;
    else
        return (char *) s1;  /* cast const away */
}
```

PE 11-9

```
/* Programming Exercise 11-9 */
#include <stdio.h>
#define LEN 81
int drop_space(char * s);
int main(void)
{
    char orig[LEN];

    while (gets(orig) && orig[0] != '\0')
    {
        drop_space(orig);
        puts(orig);
    }
    puts("Bye!");
    return 0;
}

int drop_space(char * s)
{
```

```
    int ct = 0;
    char * pos;
    while (*s)      /* or while (*s != '\0') */
    {
        if (*s == ' ')
        {
            pos = s;
            do
            {
                *pos = *(pos + 1);
                pos++;
            } while (*pos);
        }
        else
            s++;
    }
}
```

PE 11-11

```
/* pe11-11.c -- counts words and certain characters */
/* Programming Exercise 11-11                       */
#include <stdio.h>
#include <ctype.h>      // for isspace()
#include <stdbool.h>    // for bool, true, false
int main(void)
{
   char c;              // read in character
   int low_ct = 0;      // number of lowercase characters
   int up_ct = 0;       // number of uppercase characters
   int dig_ct = 0;      // number of digits
   int n_words = 0;     // number of words
   int punc_ct = 0;     // number of punctuation marks
   bool inword = false; // == true if c is in a word

   printf("Enter text to be analyzed (EOF to terminate):\n");
   while ((c = getchar()) != EOF)
   {
       if (islower(c))
          low_ct++;
       else if (isupper(c))
          up_ct++;
```

```
        else if (isdigit(c))
            dig_ct++;
        else if (ispunct(c))
            punc_ct++;
        if (!isspace(c) && !inword)
        {
            inword = true;  // starting a new word
            n_words++;      // count word
        }
        if (isspace(c) && inword)
            inword = false; // reached end of word
    }
    printf("\nwords = %d, lowercase = %d, uppercase = %d, "
           "digits = %d, punctuation = %d\n",
           n_words, low_ct, up_ct, dig_ct, punc_ct);
    return 0;
}
```

PE 11-13

```
/* Programming Exercise 11-13 */
#include <stdio.h>
#include <stdlib.h>         /* for atof()          */
#include <math.h>           /* for pow()           */
/* #include <console.h> */  /* Macintosh adjustment */
int main(int argc, char *argv[])
{
    double num, exp;

    /* argc = ccommand(&argv); */  /* Macintosh adjustment */
    if (argc != 3)
        printf("Usage: %s number exponent\n", argv[0]);
    else
    {
        num = atof(argv[1]);
        exp = atof(argv[2]);
        printf("%f to the %f power = %g\n", num, exp, pow(num,exp));
    }

    return 0;
}
```

```
/* Programming Exercise 11-15 */
#include <stdio.h>
#include <ctype.h>
/* #include <console.h> */   /* Macintosh adjustment */

int main(int argc, char *argv[])
{
    char mode = 'p';
    int ok = 1;
    int ch;

    /*argc = ccommand(&argv); */  /* Macintosh adjustment */

    if (argc > 2)
    {
        printf("Usage: %s [-p | -u | -l]\n", argv[0]);
        ok = 0;                 /* skip processing input */
    }
    else if (argc == 2)
    {
        if (argv[1][0] != '-')
        {
            printf("Usage: %s [-p | -u | -l]\n", argv[0]);
            ok = 0;
        }
        else
            switch(argv[1][1])
            {
                case 'p' :
                case 'u' :
                case 'l' : mode = argv[1][1];
                           break;
                default  : printf("%s is an invalid flag; ", argv[1]);
                           printf("using default flag (-p).\n");
            }
    }

    if (ok)
        while ((ch = getchar() ) != EOF)
        {
            switch(mode)
            {
```

```
                case 'p'  :  putchar(ch);
                             break;
                case 'u'  :  putchar(toupper(ch));
                             break;
                case 'l'  :  putchar(tolower(ch));
            }
        }

    return 0;
}
```

Chapter 12

PE 12-1

```
/* pe12-1.c  -- deglobalizing global.c */
/* Programming Exercise 12-1           */
/* one of several approaches */
#include <stdio.h>
void critic(int * u);
int main(void)
{
   int units;  /* units now local */

   printf("How many pounds to a firkin of butter?\n");
   scanf("%d", &units);
   while ( units != 56)
       critic(&units);
   printf("You must have looked it up!\n");
   return 0;
}

void critic(int * u)
{
   printf("No luck, chummy. Try again.\n");
   scanf("%d", u);
}

// or use a return value:
// units = critic();

// and have critic look like this:
```

```
/*
int critic(void)
{
   int u;
   printf("No luck, chummy. Try again.\n");
   scanf("%d", &u);
   return u;
}
*/

// or have main() collect the next value for units
```

PE 12-3

```
//pe12-3a.h

#define METRIC 0
#define US 1
#define USE_RECENT 2

void check_mode(int *pm);

void get_info(int mode, double * pd, double * pf);

void show_info(int mode, double distance, double fuel);

// pe12-3a.c
#include <stdio.h>
#include "pe12-3a.h"

void check_mode(int *pm)
{
    if (*pm != METRIC && *pm != US)
    {
        printf("Invalid mode specified. Mode %d\n", *pm);
        printf("Previous mode will be used.\n");
        *pm = USE_RECENT;
    }
}

void get_info(int mode, double * pd, double * pf)
{
    if (mode == METRIC)
```

```
        printf("Enter distance traveled in kilometers: ");
    else
        printf("Enter distance traveled in miles: ");
    scanf("%lf",pd);
    if (mode == METRIC)
        printf("Enter fuel consumed in liters: ");
    else
        printf("Enter fuel consumed  in gallons: ");
    scanf("%lf", pf);
}

void show_info(int mode, double distance, double fuel)
{
    printf("Fuel consumption is ");
    if (mode == METRIC)
        printf("%.2f liters per 100 km.\n", 100 * fuel / distance);
    else
        printf("%.1f miles per gallon.\n", distance / fuel);
}

// pe12-3.c
#include <stdio.h>
#include "pe12-3a.h"
int main(void)
{
  int mode;
  int prev_mode = METRIC;
  double distance, fuel;

  printf("Enter 0 for metric mode, 1 for US mode: ");
  scanf("%d", &mode);
  while (mode >= 0)
  {
        check_mode(&mode);
        if (mode == USE_RECENT)
            mode = prev_mode;
        prev_mode = mode;
      get_info(mode, &distance, &fuel);
      show_info(mode, distance, fuel);
        printf("Enter 0 for metric mode, 1 for US mode");
        printf(" (-1 to quit): ");
      scanf("%d", &mode);

  }
```

```
    printf("Done.\n");

    return 0;
}
```

PE 12-5

```
/* pe12-5.c  */
#include <stdio.h>
#include <stdlib.h>
void print(const int array[], int limit);
void sort(int array[], int limit);

#define SIZE 100
int main(void)
{
    int i;
    int arr[SIZE];

    for (i = 0; i < SIZE; i++)
        arr[i] = rand() % 10 + 1;
    puts("initial array");
    print(arr,SIZE);
    sort(arr,SIZE);
    puts("\nsorted array");
    print(arr,SIZE);

    return 0;
}

/* sort.c -- sorts an integer array in decreasing order */
void sort(int array[], int limit)
{
   int top, search, temp;

   for (top = 0; top < limit -1; top++)
       for (search = top + 1; search < limit; search++)
            if (array[search] > array[top])
            {
                 temp = array[search];
                 array[search] = array[top];
                 array[top] = temp;
            }
```

```
}

/* print.c -- prints an array */
void print(const int array[], int limit)
{
   int index;

   for (index = 0; index < limit; index++)
   {
      printf("%2d ", array[index]);
      if (index % 10 == 9)
         putchar('\n');
   }
   if (index % 10 != 0)
      putchar('\n');
}


PE 12-7

/* pe12-7.c  */
#include <stdio.h>
#include <stdlib.h>  /* for srand() */
#include <time.h>    /* for time()  */
int rollem(int);

int main(void)
{
    int dice, count, roll;
    int sides;
    int set, sets;

    srand((unsigned int) time(0));  /* randomize rand() */

    printf("Enter the number of sets; enter q to stop.\n");
    while ( scanf("%d", &sets) == 1)
    {
          printf("How many sides and how many dice?\n");
        scanf("%d %d", &sides, &dice);
        printf("Here are %d sets of %d %d-sided throws.\n", sets, dice,
                sides);
        for (set = 0; set < sets; set++)
        {
            for (roll = 0, count = 0; count < dice; count++)
```

```
            roll += rollem(sides);
                /* running total of dice pips */
            printf("%4d ", roll);
            if (set % 15 == 14)
                putchar('\n');
        }
        if (set % 15 != 0)
            putchar('\n');
        printf("How many sets? Enter q to stop.\n");
    }
    printf("GOOD FORTUNE TO YOU!\n");
    return 0;
}

int rollem(int sides)
{
    int roll;

    roll = rand() % sides + 1;
    return roll;
}
```

Chapter 13

PE 13-2

```
/* Programming Exercise 13-2 */
#include <stdio.h>
#include <stdlib.h>
//#include <console.h>    /* Macintosh adjustment */


int main(int argc, char *argv[])
{
    int byte;
    FILE * source;
    FILE * target;

//    argc = ccommand(&argv);    /* Macintosh adjustment */

    if (argc != 3)
    {
```

```
        printf("Usage: %s sourcefile targetfile\n", argv[0]);
        exit(EXIT_FAILURE);
    }


    if ((source = fopen(argv[1], "rb")) == NULL)
    {
        printf("Could not open file %s for input\n", argv[1]);
        exit(EXIT_FAILURE);
    }
    if ((target = fopen(argv[2], "wb")) == NULL)
    {
        printf("Could not open file %s for output\n", argv[2]);
        exit(EXIT_FAILURE);
    }
    while ((byte = getc(source)) != EOF)
    {
        putc(byte, target);
    }
    if (fclose(source) != 0)
        printf("Could not close file %s\n", argv[1]);
    if (fclose(target) != 0)
        printf("Could not close file %s\n", argv[2]);

    return 0;
}
```

PE 13-4

```
/* Programming Exercise 13-4 */
#include <stdio.h>
#include <stdlib.h>
#include <console.h>    /* Macintosh adjustment */


int main(int argc, char *argv[])
{
    int byte;
    FILE * source;
    int filect;

    argc = ccommand(&argv);    /* Macintosh adjustment */
```

```
    if (argc == 1)
    {
        printf("Usage: %s filename[s]\n", argv[0]);
        exit(EXIT_FAILURE);
    }

    for (filect = 1; filect < argc; filect++)
    {
        if ((source = fopen(argv[filect], "r")) == NULL)
        {
            printf("Could not open file %s for input\n", argv[filect]);
            continue;
        }
        while ((byte = getc(source)) != EOF)
        {
            putchar(byte);
        }
        if (fclose(source) != 0)
            printf("Could not close file %s\n", argv[1]);
    }

    return 0;
}
```

PE 13-5

```
/* Programming Exercise 13-5 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
//#include <console.h>    /* Macintosh adjustment */

#define BUFSIZE 1024
#define SLEN 81
void append(FILE *source, FILE *dest);

int main(int argc, char *argv[])
{
    FILE *fa, *fs;
    int files = 0;
    int fct;
```