1. State the data type of each of the following:
   1. character
   2. integer
   3. float
   4. character
2. a. AAAA

b. A65AA

c. AA)A

d. AAAe

1. a. The program will wait for the input of two integers separated by one space (or one white space, i.e. space, tab or newline), e.g. 23 45.

b. This is an error by omitting the ‘&’, i.e. the address operator. The program will terminate abnormally.

c. The program will wait for the input of two integers separated by one ‘/’, e.g. 23/45

1. The value of B is -0.000001.

B is assigned a value which is mathematically zero. But on most machines, this value will not be zero, showing that even the double precision is not sufficient. When a very large number (A \* A \* A) is added to a very small number (0.37 / A), the result is an approximation of the real sum. In this case the approximation is the very large number that we started with. Thus the subtraction gets the results down to zero, and the final value assigned to B is just -0.37/A. This effect is called roundoff error.

1. a. 5 \* 2 / 2 = 5

b. 5 / 2 \* 2 = 4

c. (1.2 + 10) \* 20 = 11.2 \* 20 = 224.0

d. (2++) \* 5 = 2 \* 5 = 10

e. 2++ \* 5 = 2 \* 5 = 10

f. -12L \* (3.4 – 1.2) = 1717986919

g. m = n = --2 = 1

h. (int) 3.4 \* 10 = 3 \* 10 = 30

i. (int) (3.4 \* 10) = (int) 34.0 = 34

j. j = 2 + 1.2 = (int) 3.2 = 3

1. a.
2. a. change the constant definition in #define to a lower value;

b. add an outer for loop:

for (t = 1; t <= 1000; t++)

for (count = -DLENGTH; …)

;

1. first -= second;

second += first;

first = second - first