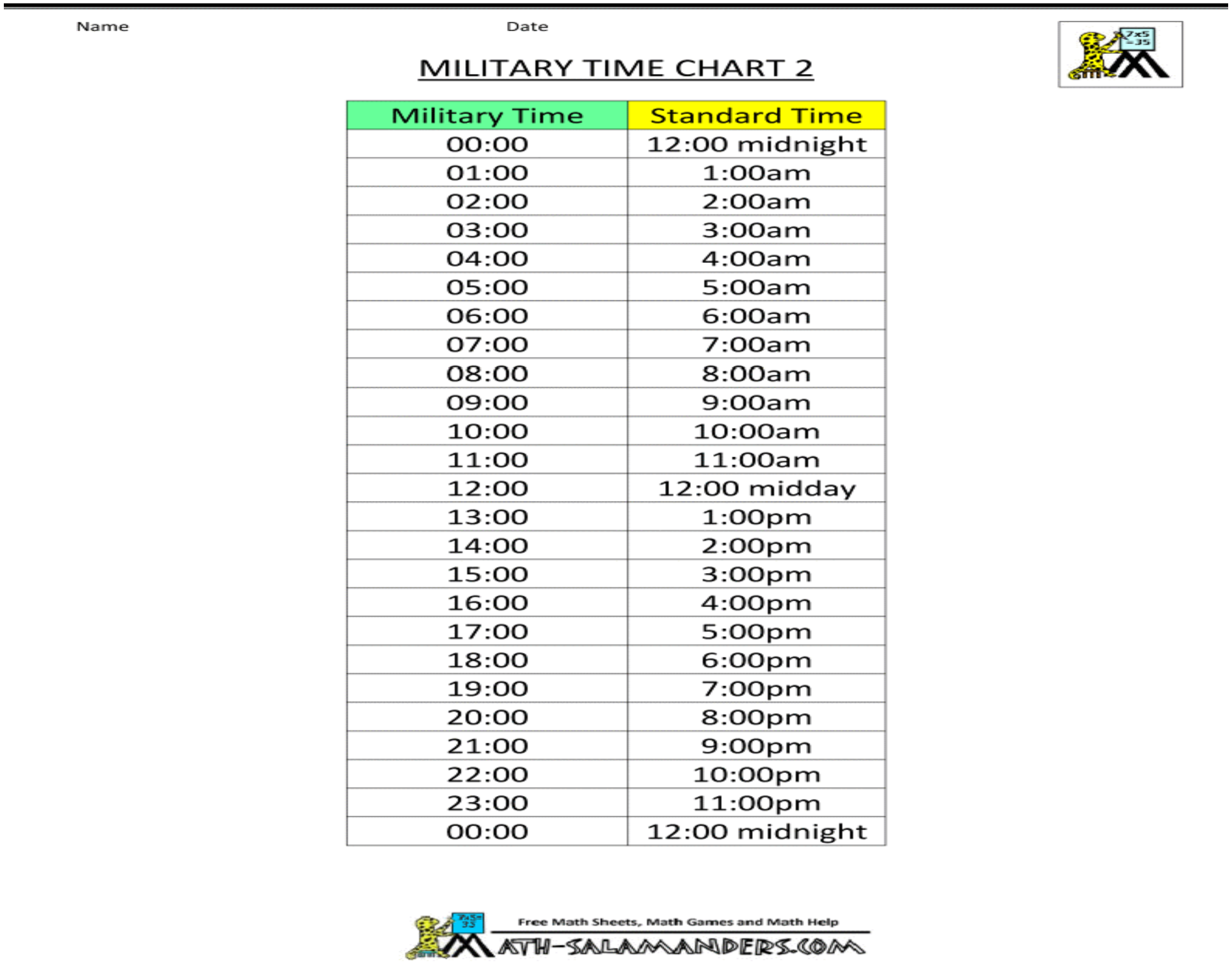
Military time to standard time



Name Date

MILITARY TIME CHART 2

Military Time	Standard Time
00:00	12:00 midnight
01:00	1:00am
02:00	2:00am
03:00	3:00am
04:00	4:00am
05:00	5:00am
06:00	6:00am
07:00	7:00am
08:00	8:00am
09:00	9:00am
10:00	10:00am
11:00	11:00am
12:00	12:00 midday
13:00	1:00pm
14:00	2:00pm
15:00	3:00pm
16:00	4:00pm
17:00	5:00pm
18:00	6:00pm
19:00	7:00pm
20:00	8:00pm
21:00	9:00pm
22:00	10:00pm
23:00	11:00pm
00:00	12:00 midnight

Increment and Decrement Operators

- Increment operator **`(++)`** can be used instead of **`c=c+1`**
- Decrement operator **`(--)`** can be used instead of **`c=c-1`**.
- Preincrement
  - Operator is used before the variable (**`++c`** or **`--c`**)
  - Variable is changed, then the expression it is in is evaluated
- Postincrement
  - Operator is used after the variable (**`c++`** or **`c--`**)
  - Expression executes, then the variable is changed

Increment and Decrement Operators (II)

- When variable not in an expression
  - Preincrementing and postincrementing have the same effect.

```
c = 20;
++c;
printf(“%d”,c);
```

and

```
 c = 20;
 c++;
 printf(“%d”,c);
```

have the same effect.

both will print 21;

Increment and Decrement Operators (III)

- When variable in an expression
  - Pre-incrementing and post-incrementing DOES NOT have the same effect.
  - Preincrement updates the variable first then evaluates expression
  - Postincrement evaluates the expression first then updates the variable

```
c = 5;
printf( "%d", ++c);          Prints 6
printf( "%d", c++);          Prints 5
```

In either case, **`c`** now has the value of **6**

Little Quiz for you (what is the output)

```
int a, b, c;
b = 10;
c = 20;
a = b+++--c;
printf("%d %d %d", a, b, c);
```

(a) Compilation error

(b) 30 10 20

(c) 30 11 19

(d) 29 11 19 OK

(e) 29 10 20

Assignment Operators (shorthand notations)

- Assignment operators abbreviate assignment expressions

  ```
  c = c + 3;
  ```

  can be abbreviated as `c += 3;` using the addition assignment operator

- Statements of the form

  variable **=** *variable operator expression*;

  can be rewritten as

  variable operator= *expression*;

- Examples of other assignment operators:

  ```
  d -= 4       (d = d - 4)
  e *= 5       (e = e * 5)
  f /= 3       (f = f / 3)
  g %= 9       (g = g % 9)
  ```

Type conversion

- Lower to higher auto-conversion (called auto-casting)

```
int x = 9;

float y = x;   //OK no warning no error
```

- Higher to lower still auto-casting but generates warning

```
float x = 9.5;

int y = x;   //OK but generates warning but no error

int y = (int) x // No warning called casting
```

- Work out the followings:

```
float x = 5/3;

int y = 5/3;
```

```
float x = 5.0/3;

int y = 5.0/3;
```

```
x = 1.0  y = 1
```

```
x = 1.6667  y = 1
```

Type conversion (example)

Floor(x) $\lfloor x \rfloor$: The largest integer not exceeding x

Ceil(x) $\lceil x \rceil$: The smallest integer not less than x

Round(x) $[x]$: The nearest integer (in case of tie take greater one)

According to the above definition when x = 2.3,

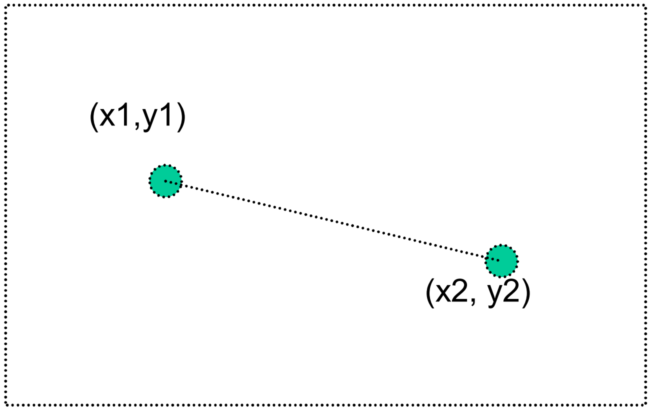
floor(x) = 2, ceil(x) = 3 and round(x) = 2

Write down a program that will take a positive fractional number as input and will print its floor, ceil and round.

Problem Solving Methodology

1. **State the problem clearly**
2. **Describe the input/output information**
3. **Work the problem by hand, give example**
4. **Develop a solution** (Algorithm Development) **and Convert it to a program** (C program)
5. **Test the solution with a variety of data**

Working with fractions: Example 2

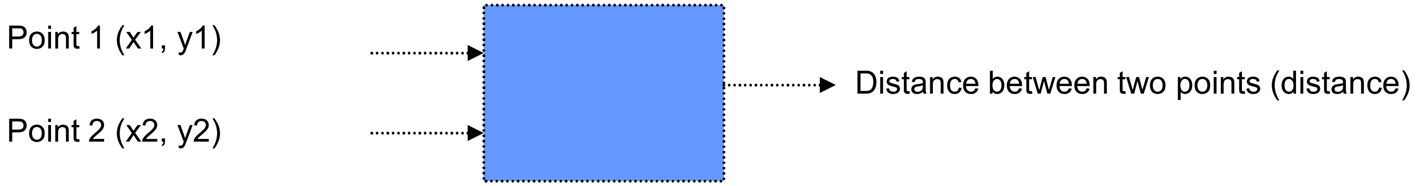


1. Problem statement

Compute the straight line

distance between two points in a plane

2. Input/output description



Example 2 (cont’d)

3. Hand example

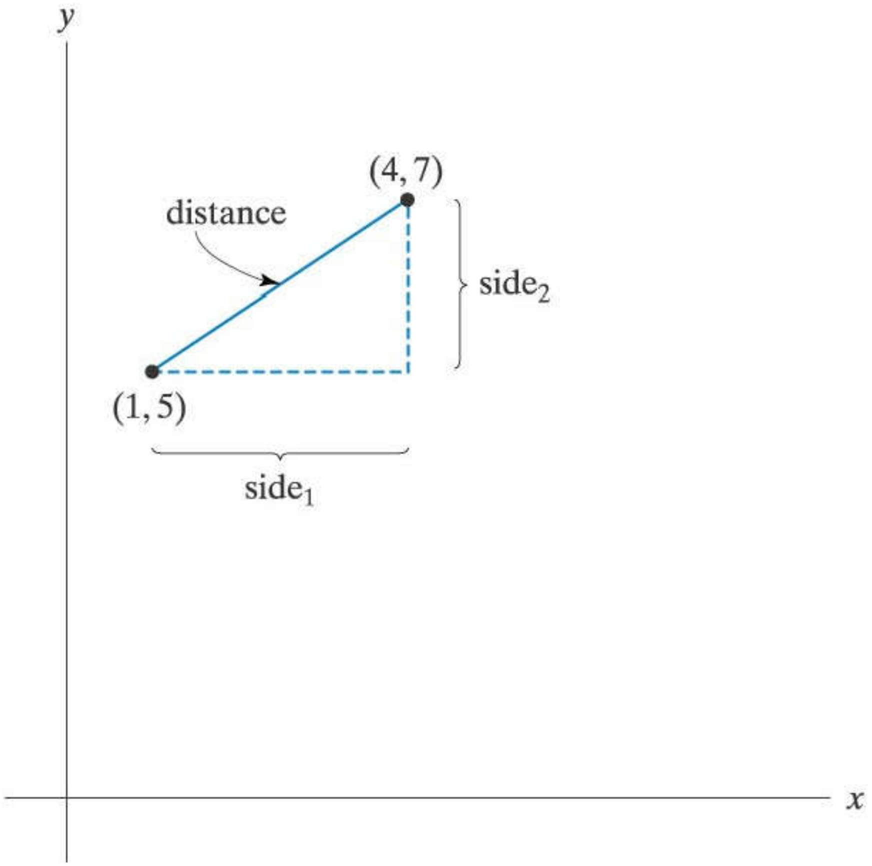
$$\text{side1} = 4 - 1 = 3$$

$$\text{side2} = 7 - 5 = 2$$

$$\text{distance} = \sqrt{\text{side1}^2 + \text{side2}^2}$$

$$\text{distance} = \sqrt{3^2 + 2^2}$$

$$\text{distance} = \sqrt{13} = 3.61$$



Example 2 (cont'd)

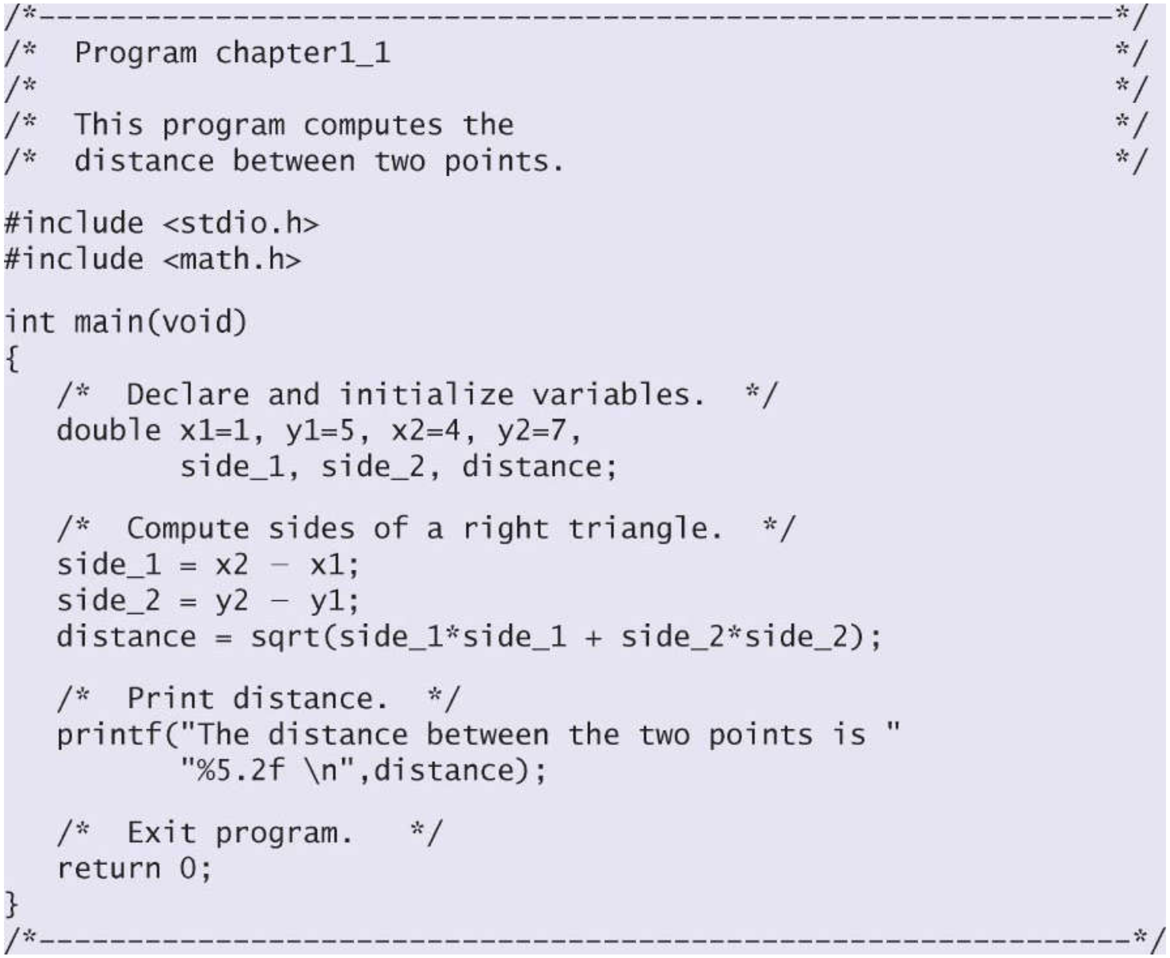
4. Algorithm development and coding

a. Generalize the hand solution and list/outline the necessary operations step-by-step

1) Give specific values for point1 (x1, y1) and point2 (x2, y2)
2) Compute side1=x2-x1 and side2=y2-y1
3) Compute $\text{distance} = \sqrt{\text{side1}^2 + \text{side2}^2}$
4) Print distance

b. Convert the above outlined solution to a program using any language you want (see next slide for C imp.)

Example 2 (cont’d)



```
/*-------------------------------------------------------------*/
/*  Program chapter1_1                                          */
/*                                                              */
/*  This program computes the                                   */
/*  distance between two points.                                */

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

int main(void)
{
   /*  Declare and initialize variables.  */
   double x1=1, y1=5, x2=4, y2=7,
          side_1, side_2, distance;

   /*  Compute sides of a right triangle.  */
   side_1 = x2 - x1;
   side_2 = y2 - y1;
   distance = sqrt(side_1*side_1 + side_2*side_2);

   /*  Print distance.  */
   printf("The distance between the two points is "
          "%5.2f \n",distance);

   /*  Exit program.   */
   return 0;
}
/*-------------------------------------------------------------*/
```

Example 2 (cont'd)

5. Testing

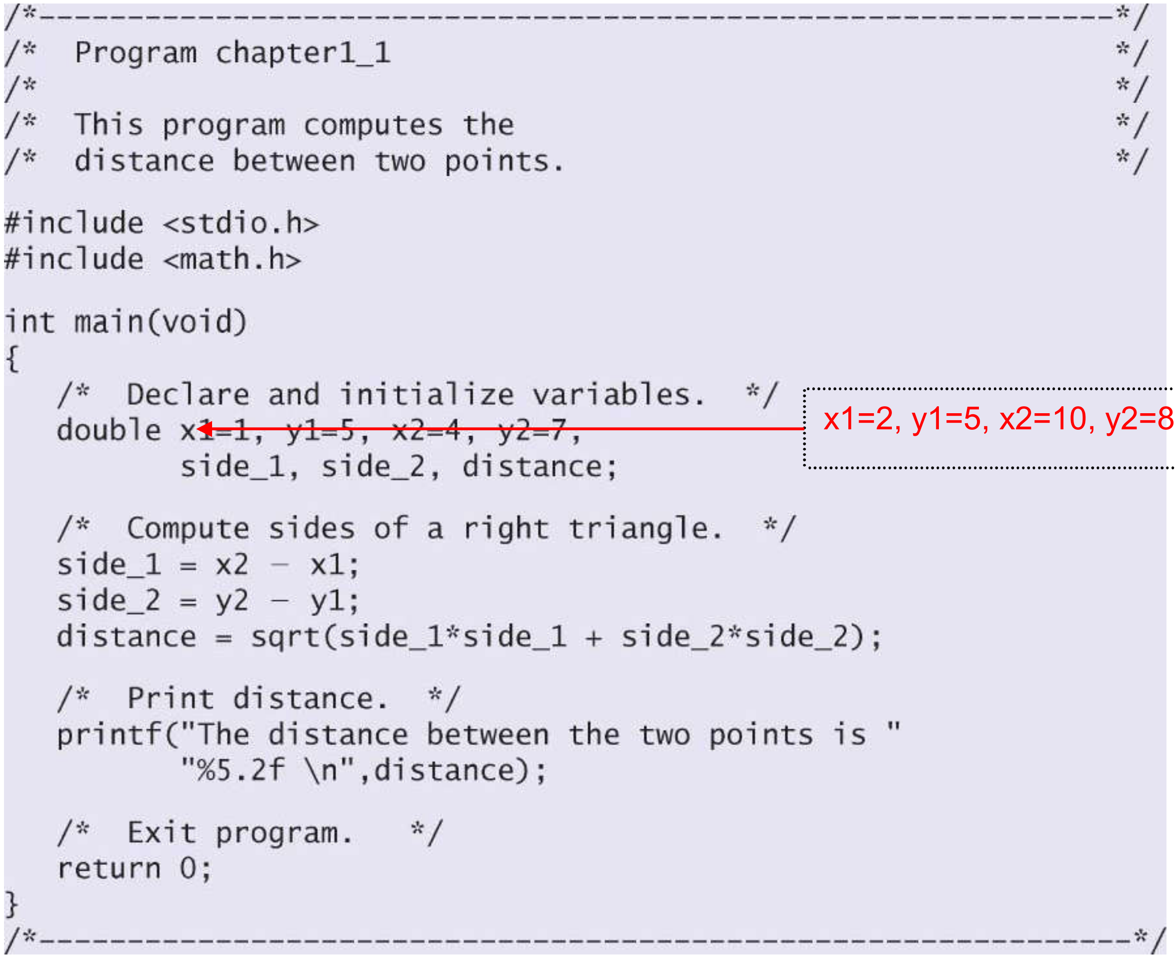
- After compiling your program, run it and see if it gives the correct result.
- Your program should print out

```
    The distance between two points is 3.61
```

- If not, what will you do?

Modification to Example 2

How will you find the distance between two other points (2,5) and (10,8)?



```
/*-------------------------------------------------------------*/
/*   Program chapter1_1                                         */
/*                                                              */
/*   This program computes the                                  */
/*   distance between two points.                               */

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

int main(void)
{
   /*  Declare and initialize variables.  */
   double x1=1, y1=5, x2=4, y2=7,
          side_1, side_2, distance;

   /*  Compute sides of a right triangle.  */
   side_1 = x2 - x1;
   side_2 = y2 - y1;
   distance = sqrt(side_1*side_1 + side_2*side_2);

   /*  Print distance.  */
   printf("The distance between the two points is "
          "%5.2f \n",distance);

   /*  Exit program.   */
   return 0;
}
/*--------------------------------------------------------------*/
```

x1=2, y1=5, x2=10, y2=8,