C++ Parameters and Overloading Practice

Computer Programming for Engineers

Week 4
Problem #1 Time Conversion

Description:
● Program that converts from 24 hour system to 12 hour system
  ○ example) It should convert 14:25 to 2:25 P.M.
● Input
  ○ 1st line: The number of test cases
  ○ The other lines: Test cases as two integers separated by colon(:)
● Output
  ○ Converted results for each test case
  ○ One line for one result
● Program should contain 3 functions
  ○ For input
  ○ For conversion
    ▪ Record the A.M./P.M. information as a value of type char, ‘A’ for A.M. and ’P’ for P.M.
    ▪ Use call-by-reference formal parameter of type char for it
  ○ For output
Problem #1 Input & Output Example

• Input
  > 3
  1:10
  20:6
  13:48

• Output
  1:10 AM
  8:6 PM
  1:48 PM
Problem #2 Arctangent Extension

Description:

- Existing `atan2` function in `math.h` is not sufficient to provide the result we want
  - The returned value for input coordinates is represented in Radian
- Extend this function for more functionalities using overloading
  - First function: gets not only the X,Y coordinates but also the phase angle parameter
  - Second function: gets only the degree value, without the X,Y coordinates
- Make sure that when the angle is greater than 360 degrees, the value proceeds back to zero
- These new functions return values in Radian, being compatible with the `atan2` function
Problem #2 Programming Conditions

● Condition
  ○ Use the given skeleton code to create the overloading functions
  ○ Functions should include default parameters
  ○ In the skeleton code, do NOT modify the codes related in standard output
  ○ Use the macro `M_PI` in `math.h` for the value of circle rate ($\pi=3.14...$).

● Input
  ○ 1st line: The number of test cases
  ○ The other lines: Test cases as follows
    ▪ Mode(c or d) + Parameters(locations, degrees, ... ) for the chosen mode

● Output
  ○ Print the function name every time `atan2` function is called
  ○ Print the radian value for the given inputs
#include <iostream>
#include <math.h>
using namespace std;

double atan2(/*Parameter Definition for Overloading*/){
    cout<<"__func__"<<": ";
    /* Implement your algorithm here*/
    return 0.0
}

int main(){
    int N, n;
    double result;
    cin>>N;
    for(n=0;n<N;n++){
        /* Implement your algorithm here */
        cout<<result<<endl;
    }

    return 0;
}
Problem #2 Input & Output Example

● Input
  > 5
  c 10.0 10.0 2.4
  c 1.0 0.0
  c -1.0 0.0 180
  d 370
  d 90

● Output
  atan2 : 1.5708
  atan2 : 0
  atan2 : 0
  atan2 : 6.28159
  atan2 : 0
  atan2 : 1.5708