3.3.15

Shivam Sawarkar
AI25BTECH11031

September 28, 2025

Question

Construct a triangle ABC in which $BC = 7cm$, and median $AD = 5cm$, $\angle A = 60^\circ$. Write the steps of construction also.

Solution

$$\mathbf{B} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \qquad \mathbf{C} = \begin{pmatrix} 7 \\ 0 \end{pmatrix}, \qquad \mathbf{D} = \begin{pmatrix} 3.5 \\ 0 \end{pmatrix}. \tag{1}$$

Since $AD = 5$, point $\mathbf{A}$ lies on the circle with center $\mathbf{D}$ and radius 5. We parametrize:

$$\mathbf{A} = \mathbf{D} + 5 \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} = \begin{pmatrix} 3.5 + 5\cos\theta \\ 5\sin\theta \end{pmatrix}. \tag{2}$$

Define the vectors

$$\mathbf{c} = \mathbf{AB} = \mathbf{B} - \mathbf{A} = \begin{pmatrix} -3.5 - 5\cos\theta \\ -5\sin\theta \end{pmatrix}, \tag{3}$$

$$\mathbf{b} = \mathbf{AC} = \mathbf{C} - \mathbf{A} = \begin{pmatrix} 3.5 - 5\cos\theta \\ -5\sin\theta \end{pmatrix}. \tag{4}$$

Solution

Angle condition:

$$\mathbf{c}^\top \mathbf{b} = \|c\| \, \|b\| \cos 60^\circ = \frac{1}{2} \|c\| \, \|b\| \tag{5}$$

Compute the dot product:

$$\mathbf{c}^\top \mathbf{b} = (-3.5 - 5\cos\theta)(3.5 - 5\cos\theta) + (-5\sin\theta)(-5\sin\theta) = \frac{51}{4}. \tag{6}$$

Hence

$$\|b\| \, \|c\| = \frac{51}{2}. \tag{7}$$

Now,

$$\|c\|^2 = \frac{149}{4} + 35\cos\theta, \qquad \|b\|^2 = \frac{149}{4} - 35\cos\theta. \tag{8}$$

Therefore,

$$(\|c\| \, \|b\|)^2 = \left(\frac{149}{4}\right)^2 - (35\cos\theta)^2. \tag{9}$$

Solution

Substituting $\|c\| \; \|b\| = \dfrac{51}{2}$,

$$\left(\frac{51}{2}\right)^2 = \left(\frac{149}{4}\right)^2 - (35\cos\theta)^2, \tag{10}$$

$$\cos^2\theta = \frac{11797}{19600}. \tag{11}$$

Thus,

$$\cos\theta = \pm\frac{\sqrt{11797}}{140}, \qquad \sin\theta = \pm\frac{\sqrt{7803}}{140}. \tag{12}$$

Finally, coordinates of A are

$$\mathbf{A} = \begin{pmatrix} \frac{7}{2} \pm \frac{\sqrt{11797}}{28} \\ \pm\frac{\sqrt{7803}}{28} \end{pmatrix}. \tag{13}$$

Plot

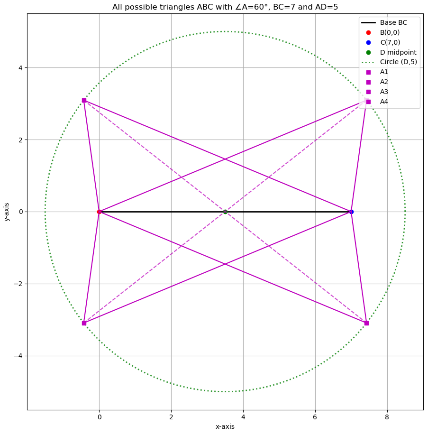
All possible triangles ABC with ∠A=60°, BC=7 and AD=5
Base BC
B(0,0)
C(7,0)
D midpoint
Circle (D,5)
A1
A2
A3
A4
x-axis
y-axis

Figure:

C Code

```
#ifndef TRIANGLE_H
#define TRIANGLE_H

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

// Struct for a point
typedef struct {
    double x, y;
} Point;
// Function to compute intersections of two circles
__attribute__((visibility("default")))
int circle_intersections(Point c1, double r1, Point c2, double
    double dx = c2.x - c1.x;
    double dy = c2.y - c1.y;
    double d = hypot(dx, dy);
```

C Code

```
if (d > r1 + r2 || d < fabs(r1 - r2) || d == 0) {
    return 0; // no intersection
}

double a = (r1*r1 - r2*r2 + d*d) / (2*d);
double h = sqrt(fmax(r1*r1 - a*a, 0));
double xm = c1.x + a*dx/d;
double ym = c1.y + a*dy/d;
double rx = -dy * (h/d);
double ry = dx * (h/d);

p1->x = xm + rx;
p1->y = ym + ry;
p2->x = xm - rx;
p2->y = ym - ry;

return 2; // two intersections
```

C Code

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

int main() {
    double BC, AD, angleA;
    printf("Enter length BC: ");
    scanf("%lf", &BC);
    printf("Enter length of median AD: ");
    scanf("%lf", &AD);
    printf("Enter angle A (in degrees): ");
    scanf("%lf", &angleA);

    Point B = {0, 0};
    Point C = {BC, 0};
    Point D = {(B.x + C.x)/2, (B.y + C.y)/2};
```

C Code

```
// Circle with center D and radius AD
    double r1 = AD;

    // Angle condition: circumcircle with radius BC/(2*sinA)
    double A_rad = angleA * M_PI / 180.0;
    double R = BC / (2.0 * sin(A_rad));

    // Two possible centers of the angle-locus circle
    Point O1 = {D.x, D.y + R};
    Point O2 = {D.x, D.y - R};

    Point p1, p2;

    printf("\nPossible coordinates of A:\n");
```

C Code

```
    // Check intersections for O1
    if (circle_intersections(D, r1, O1, R, &p1, &p2)) {
        printf("A1 = (%.3f, %.3f)\n", p1.x, p1.y);
        printf("A2 = (%.3f, %.3f)\n", p2.x, p2.y);
    }

    // Check intersections for O2
    if (circle_intersections(D, r1, O2, R, &p1, &p2)) {
        printf("A3 = (%.3f, %.3f)\n", p1.x, p1.y);
        printf("A4 = (%.3f, %.3f)\n", p2.x, p2.y);
    }

    return 0;
}
```

Python Code

```
import math

class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

def circle_intersections(c1, r1, c2, r2):
    dx = c2.x - c1.x
    dy = c2.y - c1.y
    d = math.hypot(dx, dy)

    if d > r1 + r2 or d < abs(r1 - r2) or d == 0:
        return []  # no intersection
```

Python Code

```
    a = (r1*r1 - r2*r2 + d*d) / (2*d)
    h = math.sqrt(max(r1*r1 - a*a, 0))
    xm = c1.x + a*dx/d
    ym = c1.y + a*dy/d
    rx = -dy * (h/d)
    ry = dx * (h/d)

    p1 = Point(xm + rx, ym + ry)
    p2 = Point(xm - rx, ym - ry)

    return [p1, p2]

# Input
BC = float(input("Enter length BC: "))
AD = float(input("Enter length of median AD: "))
angleA = float(input("Enter angle A (in degrees): "))
```

Python Code

```
# Points B, C, D
B = Point(0, 0)
C = Point(BC, 0)
D = Point((B.x + C.x) / 2, (B.y + C.y) / 2)

# Circle 1: center D, radius AD
r1 = AD

# Angle condition: circumcircle with radius BC/(2*sinA)
A_rad = math.radians(angleA)
R = BC / (2.0 * math.sin(A_rad))

# Two possible centers of angle-locus circle
O1 = Point(D.x, D.y + R)
O2 = Point(D.x, D.y - R)
```

Python Code

```
print("\nPossible coordinates of A:")

# Intersections with O1 circle
sol1 = circle_intersections(D, r1, O1, R)
for i, p in enumerate(sol1, start=1):
    print(f"A{i} = ({p.x:.3f}, {p.y:.3f})")

# Intersections with O2 circle
sol2 = circle_intersections(D, r1, O2, R)
for i, p in enumerate(sol2, start=len(sol1)+1):
    print(f"A{i} = ({p.x:.3f}, {p.y:.3f})")

if __name__ == "__main__":
    main()
```

Python + C Code

```
import ctypes
import math

# Load the shared library
lib = ctypes.CDLL("./solution.so")

# Define Point struct for Python
class Point(ctypes.Structure):
    _fields_ = [("x", ctypes.c_double),
                ("y", ctypes.c_double)]

# Set argument and return types
lib.circle_intersections.argtypes = [Point, ctypes.c_double, P
                                     ctypes.POINTER(Point), ct
lib.circle_intersections.restype = ctypes.c_int
```

Python + C Code

```
def circle_intersections(c1, r1, c2, r2):
    p1, p2 = Point(), Point()
    count = lib.circle_intersections(c1, r1, c2, r2,
                                     ctypes.byref(p1), ctypes.
    if count == 2:
        return [p1, p2]
    return []

def main():
    BC = float(input("Enter length BC: "))
    AD = float(input("Enter length of median AD: "))
    angleA = float(input("Enter angle A (in degrees): "))

    B = Point(0, 0)
    C = Point(BC, 0)
    D = Point((B.x + C.x) / 2, (B.y + C.y) / 2)
```

```
r1 = AD
A_rad = math.radians(angleA)
R = BC / (2.0 * math.sin(A_rad))

O1 = Point(D.x, D.y + R)
O2 = Point(D.x, D.y - R)

print("\nPossible coordinates of A:")

sol1 = circle_intersections(D, r1, O1, R)
for i, p in enumerate(sol1, start=1):
    print(f"A{i} = ({p.x:.3f}, {p.y:.3f})")
```

Python + C Code

```
    sol2 = circle_intersections(D, r1, O2, R)
    for i, p in enumerate(sol2, start=len(sol1)+1):
        print(f"A{i} = ({p.x:.3f}, {p.y:.3f})")

if __name__ == "__main__":
    main()
```