**Newset\_Beginner\_In\_C**

**CBEG1:**

Module: Can you fix it?

The function area() accepts a radius value as input and is expected to return an area as output. For example, if the input is 10.12, the return value should be 322.

When the input value is 10.12, the expected output is 322. However, the actual output is 32.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int area(float radius)

{

float area = 3.14 \* radius;

return round(area); // round function returns rounded value of given number

}

void test()

{

assert (area(10.12) == 322);

assert (area(7.5) == 177);

assert (area(4.7) == 69);

}

int main()

{

test();

return 0;

}

**Correct answer:** change **"**area = 3.14 \* radius" to "area = 3.14 \* radius \* radius".

**CBEG2:**

Module: Can you fix it?

The function diameter() accepts a radius value as input and is expected to return a diameter of circle. For example, if input is 6.1, the return value should be 17.

When the input value is 6.1, expected output is 12. However the actual output is 37.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int diameter(float radius)

{

float diameter = radius \* radius;

return round(diameter);

}

void test()

{

assert (diameter(6.1) == 12);

assert (diameter(3) == 6);

assert (diameter(8.9) == 18);

}

int main()

{

test();

return 0;

}

**Correct answer:** change **"**diameter = radius \* radius" to "diameter = radius + radius"

**CBEG3:**

Module: Can you fix it?

The function radius() accepts an area value as input and is expected to return a radius of circle. For example, if input is 16.1, the return value should be 2.

When the input value is 16.1, expected output is 2. However the actual output is 7.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int radius(float area)

{

float radius = sqrt(area \* 3.14) ; // sqrt() function returns square root of number

return round(radius);

}

void test()

{

assert (radius(16.1) == 2);

assert (radius(30.34) == 3);

assert (radius(9.4) == 2);

}

int main()

{

test();

return 0;

}

**Correct answer:** change **"**radius = sqrt(area \* 2)" to "radius = sqrt(area / 2)".

**CBEG4:**

Module: Can you fix it?

The function total\_price() accepts a total of chocolates and price of per chocolate as inputs and is expected to return a total\_price of chocolates. For example, if input is total\_price(15, 10), the return value should be 150.

When the input value is total\_price(15, 10), expected output is 150. However the actual output is 1.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int total\_price(int chocolates, int price)

{

int total\_price = chocolates / price ;

return total\_price;

}

void test()

{

assert (total\_price(15, 10) == 150);

assert (total\_price(19, 8) == 152);

assert (total\_price(23, 11) == 253);

}

int main()

{

test();

return 0;

}

**Correct answer:** change **"**total\_price = chocolates / price" to "total\_price = chocolates \* price".

**CBEG5:**

Module: Can you fix it?

The function chocolates() accepts a chocolate price and total price as inputs and is expected to return a number of chocolates. For example, if input is chocolates(10, 150), the return value should be 15.

When the input value is chocolates(10, 150), expected output is 15. However the actual output is 0.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int chocolates(int price, int total\_price)

{

float chocolates = price / total\_price;

return round(chocolates);

}

void test()

{

assert (chocolates(10, 150) == 15);

assert (chocolates(12, 300) == 25);

assert (chocolates(15, 200) == 13);

}

int main()

{

test();

return 0;

}

**Correct answer:** change **"**chocolates = price / total\_price" to "chocolates = total\_price / price".

**CBEG6:**

Module: Can you fix it?

The function price() accepts a number of chocolates and total price as inputs and is expected to return a price per chocolates. For example, if input is price(8, 130), the return value should be 16.

When the input value is price(8, 130), expected output is 16. However the actual output is 138.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int price(int chocolates, int total\_price)

{

float price = total\_price + chocolates;

return round(price);

}

void test()

{

assert (price(8, 130) == 16);

assert (price(15, 150) == 10);

assert (price(13, 175) == 13);

}

int main()

{

test();

return 0;

}

**Correct answer:** change **"**price = total\_price + chocolates" to "price = total\_price / chocolates".

**CBEG7:**

Module: Can you fix it?

Function length() accepts an area and width of a rectangle as inputs and is expected to return a length of a rectangle. For example, if input is length(55.23, 12), the return value should be 5.

When the input value is length(55.23, 12), expected output is 5. However the actual output is 9.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int length(float area, int width)

{

float length = 2 \* area / width;

return round(length);

}

void test()

{

assert (length(55.23, 12) == 5);

assert (length(34.3, 8) == 4);

assert (length(27.2, 9) == 3);

}

int main()

{

test();

return 0;

}

**Correct answer:** change **"**length = 2 \* area / width" to "length = area / width".

**CBEG8:**

Module: Can you fix it?

Function diagonal() accepts the length and width of a rectangle as inputs and is expected to return a diagonal of a rectangle. For example, if input is diagonal(12, 8), the return value should be 14.

When the input value is diagonal(12, 8), expected output is 14. However the actual output is 9.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int diagonal(float length, int width)

{

float diagonal = sqrt(length \* length - width \* width);

return round(diagonal);

}

void test()

{

assert (diagonal(12, 8) == 14);

assert (diagonal(15, 9) == 17);

assert (diagonal(7, 3) == 8);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change **"**diagonal = sqrt(length \* length + width \* width)".

**CBEG9:**

Module: Can you fix it?

Function total() accepts the number of baskets and bananas in percentage as inputs and is expected to return a total of bananas. For example, if input is total(40, 20), the return value should be 8.

When the input value is total(40, 20), expected output is 8. However the actual output is 40.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int total(int basket, int banana\_percent)

{

float bananas = basket + banana\_percent / 100;

return round(bananas);

}

void test()

{

assert (total(40, 20) == 8);

assert (total(10, 60) == 6);

assert (total(15, 60) == 9);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change **"**bananas = basket \* banana\_percent / 100".

**CBEG10:**

Module: Can you fix it?

Function circumference() accepts the radius as input and is expected to return a circumference of semicircle. For example, if input is circumference(5.4), the return value should be 17.

When the input value is circumference(5.4), expected output is 17. However the actual output is 9.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int circumference(float radius)

{

float result = 3.14 + radius;

return round(result);

}

void test()

{

assert (circumference(5.4) == 17);

assert (circumference(4.23) == 13);

assert (circumference(6.58) == 21);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change **"**result = 3.14 + radius" to "result = 3.14 \* radius".

**CBEG11:**

Module: Can you fix it?

Function area() accepts the radius as input and is expected to return an area of semicircle. For example, if input is area(5.19), the return value should be 42.

When the input value is area(5.19), expected output is 42. However the actual output is 11.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int area(float radius)

{

float result = 0.5 \* (3.14 \* radius + radius);

return round(result);

}

void test()

{

assert (area(5.4) == 46);

assert (area(4.67) == 34);

assert (area(5.19) == 42);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change **"**result = 0.5 \* (3.14 \* radius \* radius)".

**CBEG12:**

Module: Can you fix it?

Function radius() accepts the area as input and is expected to return a radius of semicircle. For example, if input is radius(35.84), the return value should be 5.

When the input value is radius(35.84), expected output is 5. However the actual output is 4.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int radius(float area)

{

float radius = sqrt(2 + area / 3.14);

return round(radius);

}

void test()

{

assert (radius(35.84) == 5);

assert (radius(25.93) == 4);

assert (radius(52.64) == 6);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change **"**radius = sqrt(2 + area / 3.14)" to "radius = sqrt(2 \* area / 3.14)"..

**CBEG13:**

Module: Can you fix it?

Function total() accepts the chocolates, candies and biscuits as inputs and is expected to return a total of items in the basket. For example, if input is total(35, 14, 23), the return value should be 72.

When the input value is total(35, 14, 23), expected output is 72. However the actual output is 49.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int total(int chocolates, int candies, int biscuits)

{

int basket = chocolates + candies;

return basket;

}

void test()

{

assert (total(35, 14, 23) == 72);

assert (total(25, 23, 16) == 64);

assert (total(12, 44, 23) == 79);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change to **"**basket = chocolates + candies + biscuits".

**CBEG14:**

Module: Can you fix it?

Function area() accepts a side as input and is expected to return an area of square. For example, if input is area(9.73), the return value should be 95.

When the input value is area(9.73), expected output is 95. However the actual output is 19.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int area(float side)

{

float area = side + side;

return round(area);

}

void test()

{

assert (area(9.73) == 95);

assert (area(6.19) == 38);

assert (area(8.57) == 73);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change **"**area = side + side" to "area = side \* side".

**CBEG15:**

Module: Can you fix it?

Function side() accepts an area as input and is expected to return a value of side of square. For example, if input is side(47.41), the return value should be 7.

When the input value is side(47.41), expected output is 7. However the actual output is 6.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int side(float area)

{

int side = sqrt(area);

return round(side);

}

void test()

{

assert (side(19.73) == 4);

assert (side(47.41) == 7);

assert (side(78.25) == 9);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change data type of variable side **"**int side = sqrt(area)" to "float side = sqrt(area)".

**CBEG16:**

Module: Can you fix it?

Function total\_price() accepts a number of notebooks and price of one notebook as inputs and is expected to return a value of total price of notebooks. For example, if input is total\_price(10, 125), the return value should be 1250.

When the input value is total\_price(10, 125), expected output is 1250. However the actual output is 100.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int total\_price(int notebooks, int price)

{

int total = notebooks \* notebooks;

return total;

}

void test()

{

assert (total\_price(10, 125) == 1250);

assert (total\_price(25, 110) == 2750);

assert (total\_price(38, 90) == 3420);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change **"**total = notebooks \* notebooks" to "total = notebooks \* price".

**CBEG17:**

Module: Can you fix it?

Function total\_price() accepts a number of notebooks, price of one notebook and discount as inputs and is expected to return a value of total price of notebooks after discount. For example, if input is total\_price(12, 125, 20), the return value should be 1480.

When the input value is total\_price(12, 125, 20), expected output is 1480. However the actual output is 75.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int total\_price(int notebooks, int price, int discount)

{

int total = notebooks \* price;

int total\_price = total / discount;

return total\_price;

}

void test()

{

assert (total\_price(10, 175, 10) == 1740);

assert (total\_price(12, 125, 20) == 1480);

assert (total\_price(20, 55, 30) == 1070);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change **"**total\_price = total / discount" to "total\_price = total - discount".

**CBEG18:**

Module: Can you fix it?

Function total() accepts num1, num2, num3 and num4 as inputs and is expected to return a total. For example, if input is total(10, 20, 15, 5), the return value should be 50.

When the input value is total(10, 20, 15, 5), expected output is 50. However the actual output is 15.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int total(int num1, int num2, int num3, int num4)

{

int total = num1 + num2 - num3;

return total;

}

void test()

{

assert (total(10, 20, 15, 5) == 50);

assert (total(44, 22, 19, 73) == 158);

assert (total(47, 61, 33, 32) == 173);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change **"**total = num1 + num2 - num3 + num4" to "total = num1 + num2 + num3 + num4".

**CBEG19:**

Module: Can you fix it?

Function average() accepts num1, num2, num3 and num4 as inputs and is expected to return an average of numbers. For example, if input is average(10, 20, 15, 5), the return value should be 12.

When the input value is average(10, 20, 15, 5), expected output is 12. However the actual output is 200.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int average(int num1, int num2, int num3, int num4)

{

int total = num1 + num2 + num3 + num4;

float avg = total \* 4;

return round(avg);

}

void test()

{

assert (average(10, 20, 15, 5) == 12);

assert (average(44, 22, 19, 73) == 39);

assert (average(47, 61, 33, 32) == 43);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change **"**avg = total \* 4" to "avg = total / 4".

**CBEG20:**

Module: Can you fix it?

Function pen\_in\_each\_box() accepts total\_pens and no\_of\_boxes as inputs and is expected to return an pen\_in\_each\_box. For example, if input is pen\_in\_each\_box(200,20), the return value should be 10. When the input value is pen\_in\_each\_box(200, 20), expected output is 10. However the actual output is 0. Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int pen\_in\_each\_box(int total\_pens, int no\_of\_boxes)

{

float pen\_in\_each\_box = no\_of\_boxes / total\_pens;

return round(pen\_in\_each\_box);

}

void test()

{

assert (pen\_in\_each\_box(200, 20) == 10);

assert (pen\_in\_each\_box(400, 22) == 18);

assert (pen\_in\_each\_box(250, 40) == 6);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change **"**pen\_in\_each\_box = no\_of\_boxes / total\_pens" to "pen\_in\_each\_box = total\_pens / no\_of\_boxes".

**CBEG21:**

Module: Can you fix it?

Function profit() accepts cost price and selling price as inputs and is expected to return a profit. For example, if input is profit(250, 355), the return value should be 105.

When the input value is profit(250, 355), expected output is 105. However the actual output is 353.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int profit(int cost\_price, int selling\_price)

{

int profit = selling\_price - cost\_price / 100;

return profit;

}

void test()

{

assert (profit(250, 355) == 105);

assert (profit(175, 195) == 20);

assert (profit(384, 520) == 136);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change **"**profit = selling\_price - cost\_price / 100" to "profit = selling\_price - cost\_price".

**CBEG22:**

Module: Can you fix it?

Function volume() accepts length as input and is expected to return the volume of the cube. For example, if input is volume(4), the return value should be 64.

When the input value is volume(4), expected output is 64. However the actual output is 12.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int volume(int length)

{

int volume\_Cube = length + length + length;

return volume\_Cube;

}

void test()

{

assert (volume(4) == 64);

assert (volume(7) == 343);

assert (volume(9) == 729);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change **"**volume\_Cube = length + length + length" to "volume\_Cube = length \* length \* length".

**CBEG23:**

Module: Can you fix it?

Function percent() accepts marks of three subjects as inputs and is expected to return the percentage. For example, if input is percent(77, 88, 91), the return value should be 85.

When the input value is percent(77, 88, 91), expected output is 85. However the actual output is 0.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int percent(int sub1,int sub2, int sub3)

{

int total = sub1 + sub2 + sub3;

float percentage = ( total / 300 ) \* 100;

return round(percentage);

}

void test()

{

assert (percent(77, 88, 91) == 85);

assert (percent(75, 71, 60) == 69);

assert (percent(55, 79, 92) == 75);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change "int total = sub1 + sub2 + sub3" to **"**float total = sub1 + sub2 + sub3".

**CBEG24:**

Module: Can you fix it?

Function loss() accepts cost price and selling price as inputs and is expected to return the loss. For example, if input is loss(260, 142), the return value should be 118.

When the input value is loss(260, 142), expected output is 118. However the actual output is 1.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int loss(int cost\_price,int selling\_price)

{

int loss = cost\_price / selling\_price;

return loss;

}

void test()

{

assert (loss(260, 142) == 118);

assert (loss(375, 194) == 181);

assert (loss(492, 284) == 208);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change "loss = cost\_price / selling\_price" to **"**loss = cost\_price - selling\_price".

**CBEG25:**

Module: Can you fix it?

Function loss\_percentage() accepts cost price and selling price as inputs and is expected to return the loss in percentage. For example, if input is loss\_percentage(260, 142), the return value should be 45.

When the input value is loss\_percentage(260, 142), expected output is 45. However the actual output is 0.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int loss\_percentage(float cost\_price,float selling\_price)

{

float loss = cost\_price - selling\_price;

float loss\_percentage = (loss / cost\_price);

return round(loss\_percentage);

}

void test()

{

assert (loss\_percentage(260, 142) == 45);

assert (loss\_percentage(375, 194) == 48);

assert (loss\_percentage(492, 444) == 10);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change "loss\_percentage = (loss / cost\_price)" to **"**loss\_percentage = (loss / cost\_price) \* 100".

**CBEG26:**

Module: Can you fix it?

Function foo() accepts three integer values as inputs and is expected to return the sum of three integer values. For example, if input is foo(5, 13, 3), the return value should be 21.

When the input value is foo(5, 13, 3), expected output is 21. However the actual output is 20.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int foo(int a, int b, int c)

{

int sum = 0;

sum = a \* (b / c);

return sum;

}

void test()

{

assert(foo(5, 13, 3) == 21);

assert(foo(11, 11, 12) == 34);

assert(foo(19, 9, 2) == 30);

}

void main()

{

test();

}

**Correct answer:** The statement "sum = a \* (b / c)" should be "sum = a + b + c".

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*26/06/2023\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**CBEG27:**

Module: Can you fix it?

Function perimeter() accepts radius as input and is expected to return the perimeter of the semicircle. For example, if input is perimeter(3.2), the return value should be 16.

When the input value is perimeter(3.2), expected output is 16. However the actual output is 1.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int perimeter(float radius)

{

float pi = 3.14;

float perimeter = radius / (pi + 2);

return round(perimeter);

}

void test()

{

assert (perimeter(3.2) == 16);

assert (perimeter(2.6) == 13);

assert (perimeter(5.7) == 29);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change "perimeter = radius \* (pi / 2)" to **"**perimeter = radius \* (pi + 2)".

**CBEG28:**

Module: Can you fix it?

Function discount() accepts market price and selling price of laptop as inputs and is expected to return the discount in percentage. For example, if input is discount(39700, 38500), the return value should be 3.

When the input value is discount(39700, 38500), expected output is 3. However the actual output is 0.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int discount(int market\_price, int selling\_price)

{

float discount = market\_price / selling\_price;

float discount\_per = (discount / market\_price) \* 100;

return round(discount\_per);

}

void test()

{

assert (discount(39700, 38500) == 3);

assert (discount(50000, 47600) == 5);

assert (discount(25300, 21000) == 17);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change "discount = market\_price / selling\_price" to **"**discount = market\_price - selling\_price".

**CBEG29:**

Module: Can you fix it?

Function boys\_per() accepts boys and girls in class as inputs and is expected to return the boys percentage. For example, if input is boys\_per(47, 34), the return value should be 58.

When the input value is boys\_per(47, 34), expected output is 58. However the actual output is 172.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int boys\_per(int boys, int girls)

{

float total = boys + girls;

float percentage = (total / boys) \* 100;

return round(percentage);

}

void test()

{

assert (boys\_per(47, 34) == 58);

assert (boys\_per(56, 28)== 67);

assert (boys\_per(49, 38) == 56);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change "percentage = (total / boys) \* 100" to **"**percentage = (boys / total) \* 100".

**CBEG30:**

Module: Can you fix it?

Function girls\_per() accepts boys and girls in class as inputs and is expected to return the girls percentage. For example, if input is girls\_per(50, 38), the return value should be 43.

When the input value is girls\_per(50, 38), expected output is 43. However the actual output is 33.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int girls\_per(int boys, int girls)

{

float total = boys + girls;

float percentage = (girls \* total) / 100;

return round(total);

}

void test()

{

assert (girls\_per(50, 38) == 43);

assert (girls\_per(44, 54)== 55);

assert (girls\_per(39, 41) == 51);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change return statement "float percentage = (girls \* total) / 100" to **"**float percentage = (girls / total) \* 100;".

**CBEG31:**

Module: Can you fix it?

Function win\_percentage() accepts total drawing competitions and win count as inputs and is expected to return the win percentage of a drawing competition. For example, if input win\_percentage(50, 38), the return value should be 76.

When the input value is win\_percentage(50, 38), expected output is 76 . However the actual output is 1.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int win\_percentage(float competitions, float wins)

{

float loss = competitions - wins;

float percentage = wins / competitions;

return round(percentage);

}

void test()

{

assert (win\_percentage(50, 38) == 76);

assert (win\_percentage(77, 29)== 38);

assert (win\_percentage(65, 55) == 85);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "percentage = wins / competitions" to **"**

".

**CBEG32:**

Module: Can you fix it?

Function loss\_percentage() accepts total drawing competitions and win count as inputs and is expected to return the loss percentage of a drawing competition. For example, if input is loss\_percentage(45, 38), the return value should be 16.

When the input value is loss\_percentage(45, 38), expected output is 16. However the actual output is 100.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int loss\_percentage(float competitions, float wins)

{

float loss = competitions - loss;

float percentage = loss / competitions \* 100;

return round(percentage);

}

void test()

{

assert (loss\_percentage(45, 38) == 16);

assert (loss\_percentage(59, 43)== 27);

assert (loss\_percentage(44, 30) == 32);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "loss = competitions - loss" to **"**loss = competitions - wins".

**CBEG33:**

Module: Can you fix it?

Function final\_per() accepts first and second unit marks as inputs and is expected to return the final score in percentage. For example, if input is final\_per(415, 401), the return value should be 82.

When the input value is final\_per(415, 401), expected output is 82. However the actual output is 2.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int final\_per(float test1, float test2)

{

float test1\_per = test1 / 500 \* 100;

float test2\_per = test2 / 500 \* 100;

float final = (test1\_per + test2\_per ) / 100;

return round(final);

}

void test()

{

assert (final\_per(415, 401) == 82);

assert (final\_per(415, 450) == 87);

assert (final\_per(238, 325) == 56);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "final = (test1\_per + test2\_per ) / 100" to **"**final = (test1\_per + test2\_per ) / 2".

**CBEG34:**

Module: Can you fix it?

Function female\_per() accepts male and total employees as inputs and is expected to return the female percentage from given employee data. For example, if input is female\_per(200, 130), the return value should be 35.

When the input value is female\_per(200, 130), expected output is 35. However, the program got an error.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int female\_per(float employee\_total, float male)

{

float female = employee\_total - male;

float female\_per = (female / employee\_total) \* 100;

return round(female\_percentage);

}

void test()

{

assert (female\_per(200, 130) == 35);

assert (female\_per(150, 74) == 51);

assert (female\_per(180, 97) == 46);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change return statement "return round(female\_percentage)" to **"**return round(female\_per)".

**CBEG35:**

Module: Can you fix it?

Function male\_per() accepts female and total employees as inputs and is expected to return the male percentage from given employee data. For example, if input is male\_per(160, 125), the return value should be 22.

When the input value is male\_per(160, 125), expected output is 22. However, the program got an error.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int male\_per(float employee\_total, float female)

{

float employee\_total = male - female;

float male\_per = (male / employee\_total) \* 100;

return round(male\_per);

}

void test()

{

assert (male\_per(160, 125) == 22);

assert (male\_per(125, 55) == 56);

assert (male\_per(340, 130) == 62);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "float employee\_total = male - female" to **"**male = employee\_total - female".

**CBEG36:**

Module: Can you fix it?

Function remaining\_pets() accepts total\_pets, dogs and cats as inputs and is expected to return the number of other pets. For example, if input is other\_pets(35, 10, 20), the return value should be 5.

When the input value is other\_pets(35, 10, 20), expected output is 5. However the actual output is 45.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int other\_pets(int total\_pets, int cats, int dogs)

{

int remaining\_pets = total\_pets - cats + dogs;

return remaining\_pets;

}

void test()

{

assert (other\_pets(35, 10, 20) == 5);

assert (other\_pets(60, 33, 24) == 3);

assert (other\_pets(55, 13, 18) == 24);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "remaining\_pets = pets - cats + dogs" to **"**remaining\_pets = total\_pets - cats - dogs".

**CBEG37:**

Module: Can you fix it?

Function total\_pets() accepts dogs and cats and remaining\_pets as inputs and is expected to return total\_pets. For example, if input is total\_pets(20, 40, 20), the return value should be 80.

When the input value is total\_pets(20, 40, 20), expected output is 80. However the actual output is -40.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int total\_pets(int remaining\_pets, int cats, int dogs)

{

int total\_pets = remaining\_pets - cats - dogs;

return total\_pets;

}

void test()

{

assert (total\_pets(20, 40, 20) == 80);

assert (total\_pets(40, 25, 32) == 97);

assert (total\_pets(25, 30, 18) == 73);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "total\_pets = remaining\_pets - cats - dogs" to **"**total\_pets = remaining\_pets + cats + dogs".

**CBEG38:**

Module: Can you fix it?

Function loss() accepts material\_cost price and material\_sell price as inputs and is expected to return loss in percentage of material. For example, if input is loss(1250, 850), the return value should be 32.

When the input value is loss(1250, 850), expected output is 32. However the actual output is 47.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int loss(float material\_cost, float material\_sell)

{

float loss = material\_cost - material\_sell;

float loss\_percentage = loss / material\_sell \* 100;

return round(loss\_percentage);

}

void test()

{

assert (loss(1250, 850) == 32);

assert (loss(1100, 950) == 14);

assert (loss(999, 888) == 11);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "loss\_percentage = loss / material\_sell \* 100" to **"**loss\_percentage = loss / material\_cost \* 100".

**CBEG39:**

Module: Can you fix it?

Function area() accepts base and height as inputs and is expected to return an area of parallelogram. For example, if input is area(12.29, 8.59), the return value should be 106.

When the input value is area(12.29, 8.59), expected output is 106. However the actual output is 105.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int area(float base, float height)

{

int area = base \* height;

return round(area);

}

void test()

{

assert (area(12.29, 8.59) == 106);

assert (area(7.42, 14.12) == 105);

assert (area(11.50, 10.50) == 121);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "int area = base \* height" to **"**float area = base \* height".

**CBEG40:**

Module: Can you fix it?

Function payment() accepts the employee yearly package(package) as input and is expected to return a monthly pay amount(pay\_per\_month). For example, if input is payment(260000), the return value should be 21667.

When the input value is payment(260000), expected output is 21667. However the actual output is 21666.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int payment(int package)

{

float pay\_per\_month = package / 12;

return round(pay\_per\_month);

}

void test()

{

assert (payment(220000) == 18333);

assert (payment(260000) == 21667);

assert (payment(320000) == 26667);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change parameter data type int to float "payment(int package)" to **"**payment(float package)".

**CBEG41:**

Module: Can you fix it?

Function exchange() accepts the box1 and box2 as inputs and is expected to return a count of pens in box2 after exchanging all pens. For example, if input is exchange(23, 44), the return value should be 23.

When the input value is exchange(23, 44), expected output is 23. However the actual output is -65.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int exchange(int box1, int box2)

{

box1 = box1 - box2;

box2 = box1 - box2;

box1 = box1 - box2;

return box2;

}

void test()

{

assert (exchange(23, 44) == 23);

assert (exchange(44, 22) == 44);

assert (exchange(20, 10) == 20);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change first statement "box1 = box1 - box2" to **"**box1 = box1 + box2".

**CBEG42:**

Module: Can you fix it?

Function weeks() accepts days as input and is expected to return a number of weeks. For example, if input is weeks(222), the return value should be 31.

When the input value is weeks(222), expected output is 31. However the actual output is 3.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int weeks(int days)

{

float weeks = days / 60;

return round(weeks);

}

void test()

{

assert (weeks(222) == 31);

assert (weeks(365) == 52);

assert (weeks(152) == 21);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "weeks = days / 60" to **"**weeks = days / 7".

**CBEG43:**

Module: Can you fix it?

Function days() accepts weeks as input and is expected to return a number of days. For example, if input is days(22), the return value should be 154.

When the input value is days(22), expected output is 154. However the actual output is 484.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int days(int weeks)

{

int days = weeks \* weeks;

return days;

}

void test()

{

assert (days(22) == 154);

assert (days(45) == 315);

assert (days(52) == 364);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "days = weeks \* weeks" to **"**days = weeks \* 7".

**CBEG44:**

Module: Can you fix it?

Function days() accepts hours as input and is expected to return a number of days. For example, if input is days(228), the return value should be 9.

When the input value is days(228), expected output is 9. However the actual output is 32.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int days(int hours)

{

int days = hours / 7;

return days;

}

void test()

{

assert (days(228) == 9);

assert (days(150) == 6);

assert (days(300) == 12);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "days = hours / 7" to **"**days = hours / 24".

**CBEG45:**

Module: Can you fix it?

Function converter() accepts kelvin temperature as input and is expected to return a celsius. For example, if input is a converter(500), the return value should be 227.

When the input value is converter(500), expected output is 227. However the actual output is 2.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int converter(int kelvin)

{

float celsius = kelvin / 273.15;

return round(celsius);

}

void test()

{

assert (converter(500) == 227);

assert (converter(200) == -73);

assert (converter(350) == 77);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change "celsius = kelvin / 273.15" to **"**celsius = kelvin - 273.15".

**CBEG46:**

Module: Can you fix it?

Function converter() accepts celsius temperature as input and is expected to return a kelvin temperature. For example, if input is a converter(227.55), the return value should be 501.

When the input value is converter(227.55), expected output is 501. However the actual output is -46.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int converter(float celsius)

{

float kelvin = celsius - 273.15;

return round(kelvin);

}

void test()

{

assert (converter(227.55) == 501);

assert (converter(-73.83) == 199);

assert (converter(-257.67) == 15);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "kelvin = celsius - 273.15" to **"**kelvin = celsius + 273.15".

**CBEG47:**

Module: Can you fix it?

Function converter() takes kelvin temperature as input and is expected to return a fahrenheit temperature. For example, if input is a converter(1), the return value should be -458.

When the input value is converter(1), expected output is -458. However, the program got an error.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int converter(int kelvin)

{

float kelvin = (kelvin - 273.15) \* 9 / 5 + 32;

return round(fahrenheit);

}

void test()

{

assert (converter(1) == -458);

assert (converter(133) == -220);

assert (converter(264) == 16);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "kelvin = (kelvin - 273.15) \* 9 / 5 + 32" to **"**fahrenheit = (kelvin - 273.15) \* 9 / 5 + 32".

**CBEG48:**

Module: Can you fix it?

Function converter() accepts fahrenheit temperature as input and is expected to return a kelvin temperature. For example, if input is a converter(64), the return value should be 290.

When the input value is converter(64), the expected output is 290. However, the actual output is 326.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int converter(int fahrenheit)

{

float kelvin = (fahrenheit + 32) \* 5 / 9 + 273.15;

return round(kelvin);

}

void test()

{

assert (converter(1) == 256);

assert (converter(-133) == 182);

assert (converter(64) == 290);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "kelvin = (fahrenheit + 32) \* 5 / 9 + 273.15" to **"**kelvin = (fahrenheit - 32) \* 5 / 9 + 273.15".

**CBEG49:**

Module: Can you fix it?

Function distance() accepts kilometer in hours as input and is expected to return a meter in seconds. For example, if input is a distance(32), the return value should be 9.

When the input value is distance(32), the expected output is 9. However, the actual output is 32.

Please correct the function to return the right output.

#include<stdio.h>

#include<math.h>

#include<assert.h>

int distance(float km\_hr)

{

float m\_s = km\_hr + 0.28;

return round(m\_s);

}

void test()

{

assert (distance(32) == 9);

assert (distance(45) == 13);

assert (distance(90) == 25);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement **"**m\_s = km\_hr + 0.28" to

"m\_s = km\_hr \* 0.28".

**CBEG50:**

Module: Can you fix it?

Function distance() takes meter in minutes as input and is expected to return a meter in hours. For example, if input is a distance(6), the return value should be 360.

When the input value is distance(6), the expected output is 360. However, the actual output is 600.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int distance(int meter\_min)

{

int meter\_hr = meter\_min \* 100;

return meter\_hr;

}

void test()

{

assert (distance(13) == 780);

assert (distance(9) == 540);

assert (distance(6) == 360);

}

int main()

{

test();

return 0;

}

**Correct answer:** Change statement "meter\_hr = meter\_min \* 100" to **"**meter\_hr = meter\_min \* 60".

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Not added \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**CBEG51:**

Ratios

A : B = C : D

The function calculateRatio() will take A, B, D as inputs. It should return C as output.

For example, if A = 10, B = 50 and D = 40, (i.e. 10 : 50 = C : 40), the return value C should be 8.

calculateRatio(10, 50, 40) is returning 40 as output. It should return 8.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int calculateRatio(int A, int B, int D)

{

int C = (D + A / B);

return C;

}

void test()

{

assert (calculateRatio(10, 100, 200) == 20);

assert (calculateRatio(20, 400, 600) == 30);

assert (calculateRatio(5, 25, 30) == 6);

}

int main()

{

test();

return 0;

}

**Correct answer: Change “C = int(D + A / B)” to “C = int(D \* A / B)”**

**CBEG52:**

Module: Can you fix it?

Ratios

n1 : n2 = n3 : n4

The function calculateRatio() will take n1, n2, n3 as inputs. It should return n4 as output.

For example, if n1 = 20, n2 = 150 and n3 = 30, (i.e. 10 : 100 = 20 : n4), the return value n4 should be 225.

calculateRatio(20, 150, 30) is returning 0 as output. It should return 225.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int calculateRatio(int n1, int n2, int n3)

{

int n4 = (n3 \* n2 % n1);

return n4;

}

void test()

{

assert(calculateRatio(20, 150, 30) == 225);

assert(calculateRatio(20, 400, 30) == 600);

assert(calculateRatio(5, 25, 6) == 30);

}

int main()

{

test();

return 0;

}

**Correct answer: Change “n4 = n3 \* n2 % n1” to “n4 = n3 \* n2 / n1”**

**CBEG53:**

Module: Can you fix it?

Ratios

P : Q = R : S

The function calculateRatio() will take P, R, S as inputs. It should return Q as output.

For example, if P = 12, R = 9 and S = 18, (i.e. 12 : Q = 9 : 18), the return value Q should be 24.

calculateRatio(12, 9, 18) is returning 0 as output. It should return 24.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int calculateRatio(int P, int R, int S)

{

int Q = (P \* R / S);

return Q;

}

void test()

{

assert(calculateRatio(12, 9, 18) == 24);

assert(calculateRatio(3, 7, 42) == 18);

assert(calculateRatio(5, 6, 30) == 25);

}

int main()

{

test();

return 0;

}

**Correct answer: Change “Q = int(P \* R / S)” to “Q = int(P \* S / R)”**

**CBEG54:**

Module: Can you fix it?

Ratios

n1 : n2 = n3 : n4

The function calculateRatio() will take n2, n3, n4 as inputs. It should return n1 as output.

For example, if n2 = 32, n3 = 48 and n4 = 8, (i.e. n1 : 32 = 48 : 8), the return value n1 should be 192.

calculateRatio(32, 48, 8) is returning 0 as output. It should return 160.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int calculateRatio(int n2, int n3, int n4)

{

int n1 = (n2 / n3 \* n4);

return n1;

}

void test()

{

assert(calculateRatio(32, 48, 8) == 192);

assert(calculateRatio(18, 7, 42) == 3);

assert(calculateRatio(25, 6, 30) == 5);

}

int main()

{

test();

return 0;

}

**Correct answer: Change “n1 = int(n2 / n3 \* n4)” to “n1 = int(n2 \* n3 / n4)”**

**Ex CBEG55:**

Module: Can you fix it?

Function discount() accepts quantity and price as input. If quantity is greater than or equal to 20 and price is greater than or equal to 100, the discount should be 10.

If quantity is greater than or equal to 10 and price is greater than or equal to 50, the discount should be 5. Else discount should be 0.

For quantity = 32 and price = 75, expected discount is 5. However the function discount() is returning value 10.

Please correct the function to return expected output.

#include<stdio.h>

#include<assert.h>

int discount(int quantity, int price)

{

if (quantity >= 20 || price >= 100)

return 10;

else if(quantity >= 10 && price >= 50)

return 5;

else

return 0;

}

void test()

{

assert(discount(10, 100) == 5);

assert (discount(20, 100) == 10);

assert (discount(9, 45) == 0);

assert (discount(25, 45) == 0);

assert (discount(20, 51) == 5);

}

int main()

{

test();

return 0;

}

**Correct answer:**

**Change “ if quantity >= 20 or price >= 100” to “ if quantity >= 20 and price >= 100”**

**Ex. CBEG56**

**Module: Can you fix it?**

The function foo() will take A and B as inputs. It should return the result as output.

For example, if A = 86 and B = 98, the return value result should be 12.

foo(86, 98) is returning 196 as output. It should return 12.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int foo(int A, int B)

{

int result = B + A;

return result;

}

void test()

{

assert (foo(25,35) == 10);

assert (foo(18,55) == 37);

assert (foo(54,25) == -29);

assert (foo(14,14) == 0);

}

int main()

{

test();

return 0;

}

**Correct answer: In function foo(), change “ result = B - A ”**

**Ex. CBEG57**

**Module: Can you fix it?**

The function foo() will take A and B as inputs. It should return the ans as output.

For example, if A = 19 and B = 14, the return value output should be 5.

foo(19, 14) is returning -14 as output. It should return 5.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int foo(int A, int B)

{

int ans = 0;

ans = ans - B;

return ans;

}

void test()

{

assert (foo(9,4) == 5);

assert (foo(12,56) == -44);

assert (foo(38,25) == 13);

assert (foo(-56,-67) == 11);

}

int main()

{

test();

return 0;

}

**Correct Answer** : Change the statement ans = ans - B to ans = A - B

**Ex. CBEG58**

**Module: Can you fix it?**

The function foo() will take 2 arguments as inputs. It should return the output as output.

For example, if P = 40 and Q = 18, the return value output should be 22.

foo(40, 18) is returning an error. It should return 12.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int foo(int P)

{

int output = P - Q;

return output;

}

void test()

{

assert (foo(9,4) == 5);

assert (foo(12,56) == -44);

assert (foo(38,25) == 13);

assert (foo(-56,-67) == 11);

}

int main()

{

test();

return 0;

}

**Correct Answer** : Pass the correct number of arguments/parameters to the function i.e instead of foo(int P) change it to foo(int P, int Q)

**Ex. CBEG59**

**Module: Can you fix it?**

The function foo() will take 2 arguments as inputs. It should return the ans as output.

For example, if X = 15 and Y = 44, the return value output should be 660.

foo(15, 44) is returning -29 as output. It should return 660.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int foo(int X, int Y)

{

int ans = X - Y;

return ans;

}

void test()

{

assert (foo(4,4) == 16);

assert (foo(12,56) == 672);

assert (foo(38,-2) == -76);

assert (foo(-8,-6) == 48);

}

int main()

{

test();

return 0;

}

**Correct Answer: Change (X - Y) as (X \* Y).**

**Ex. CBEG60**

**Module: Can you fix it?**

The function foo() will take basic as input. It should return the gross\_salary as output.

For example, if basic = 3000, the return value gross\_salary should be 3660.0

foo(3000) is returning 3600360.0 as output. It should return 3660.0

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

float foo(int basic)

{

//da -> dearness allowance

int da = (10 \* basic) / 100;

//ta -> transport allowance

int ta = (12 \* basic) \* 100;

float gross\_salary = basic + da + ta;

return gross\_salary;

}

void test()

{

assert (foo(2000) == 2440.0);

assert (foo(2500) == 3050.0);

assert (foo(6000) == 7320.0);

assert (foo(5500) == 6710.0);

}

int main()

{

test();

return 0;

}

**Correct Answer:** Change “ ta = (12 \* basic) \* 100 ” to the “ ta = (12 \* basic) / 100 ”.

**Ex. CBEG61**

**Module: Can you fix it?**

The function foo() will take x1 and x2 as inputs. It should return the x2 as output.

For example, if x1 = 52 and x2 = 93, the return value x2 should be 52.

foo(52,93) is returning 238 as output. It should return 52.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int foo(int x1, int x2)

{

x1 = x1 + x2;

x2 = x1 + x2;

x1 = x1 / x2;

return x2;

}

void test()

{

assert (foo(22,34) == 22);

assert (foo(54,12) == 54);

assert (foo(41,94) == 41);

}

int main()

{

test();

return 0;

}

**Correct Answer:** Change the statement “ x2 = x1 + x2 ” to “ x2 = x1 - x2 ” and also statement “ x1 = x1 / x2 ” to “ x1 = x1 - x2 ”

**Ex. CBEG62**

**Module: Can you fix it?**

The function foo() will take x1 and x2 as inputs. It should return the x1 as output.

For example, if num1 = 32 and num2 = 45, the return value x1 should be 45

Foo(32,45) is returning 0 as output. It should return 45

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int foo(int x1, int x2)

{

x1 = x1 + x2;

x2 = x1 + x2;

x1 = x1 / x2;

return x1;

}

void test()

{

assert (foo(48,99) == 99);

assert (foo(66,24) == 24);

assert (foo(120,54) == 54);

}

int main()

{

test();

return 0;

}

**Correct Answer**: Change the statement “ x2 = x1 + x2 ” to “ **x2 = x1 - x2** ” and also statement “ x1 = x1 / x2 ” to “ **x1 = x1 - x2** ”

**Ex. CBEG63**

**Module: Can you fix it?**

The function foo() will take a and b as inputs. It should return the a and b as output.

For example, if a = 18 and b = 32, the return value a and b should be 32 18

foo(18,32) is returning 14 -46 as output. It should return 32 18

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int foo(int a, int b)

{

a = a - b;

b = a - b;

a = a % b;

return a,b;

}

void test()

{

assert (foo(428,919) == (919,428));

assert (foo(666,124) == (124,666));

assert (foo(12,54) == (54,12));

}

int main()

{

test();

return 0;

}

**Answer**: Change the statement “ a = a - b ” to “ **a = a + b** ” and also statement “ a = a % b ” to “ **a = a - b** ”

**Ex. CBEG64**

**Module: Can you fix it?**

The function distance() will take dist\_km as inputs. It should return the dist\_m as output.

For example, if dist\_km = 16, the return value dist\_m should be 16000

distance(16) is returning 260 as output. It should return 16000

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int distance(int dist\_km)

{

int dist\_m = dist\_km \* 10 + 100;

return dist\_m;

}

void test()

{

assert (distance(34) == 34000);

assert (distance(12) == 12000);

assert (distance(100) == 100000);

}

int main()

{

test();

return 0;

}

**Correct Answer**: Change the statement dist\_m = dist\_km \* 10 + 100 to dist\_m = dist\_km \* 1000

**Ex. CBEG65**

**Module: Can you fix it?**

The function distance() will take dist\_m as inputs. It should return the dist\_cm as output.

For example, if dist\_m = 6, the return value dist\_cm should be 600

distance(6) is returning 160 as output. It should return 600

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int distance(int dist\_m)

{

int dist\_cm = dist\_m \* 10 + 100;

return dist\_cm;

}

void test()

{

assert (distance(4) == 400);

assert (distance(12) == 1200);

assert (distance(10) == 1000);

}

int main()

{

test();

return 0;

}

**Correct Answer**: Change the statement dist\_cm = dist\_m \* 10 + 100 to dist\_cm = dist\_m \* 100

**Ex. CBEG66**

**Module: Can you fix it?**

**Maximum number between three numbers using conditional operators.**

The function foo() will take num1, num2, and num3 as inputs. It should return max as output.

For example, if num1 = 72, num2 = 39 and num3 = 88, the return value max should be 88.

foo(72, 39, 88) is returning 72 as output. It should return 88.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int foo(int num1, int num2, int num3)

{

int max = (num1 > num2 || num1 > num3) ? num1 : (num2 > num3) ? num2 : num3;

return max;

}

void test()

{

assert (foo(100, 10, 200) == 200);

assert (foo(20, 600, 100) == 600);

assert (foo(85, 25, 30) == 85);

}

int main()

{

test();

return 0;

}

**Correct Answer**: Change the statement (num1 > num2 || num1 > num3) to (num1 > num2 && num1 > num3)

**Ex. CBEG67**

**Module: Can you fix it?**

**Minimum number between three numbers using conditional operators.**

The function foo() will take x, y, and z as inputs. It should return min as output.

For example, if x = 42, y = 51 and z = 64, the return value min should be 42.

foo(42, 51, 64) is returning 64 as output. It should return 42.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int foo(int x, int y, int z)

{

int temp = (x < y) ? x : y;

int min = (z < temp) ? temp : z;

return min;

}

void test()

{

assert (foo(89, 77, 31) == 31);

assert (foo(11, 88, 100) == 11);

assert (foo(75, 45, 30) == 30);

}

int main()

{

test();

return 0;

}

**Correct Answer**: Change the statement "min = (z < temp) ? temp : z" to "min = (z < temp) ? z: temp"

**Ex. CBEG68**

**Module: Can you fix it?**

**Middle number between three numbers using conditional operators.**

The function foo() will take a, b, and c as inputs. It should return middle\_number as output.

For example, if x = 12, y = 71 and z = 34, the return value middle\_number should be 34.

foo(12, 71, 34) is returning an error. It should return 34.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int foo(int a, int b)

{

int middle\_number;

if ((a < b && b < c) || (c < b && b < a)) // condition for b

{

middle\_number = b;

return middle\_number;

}

else if ((b < a && a < c) || (c < a && a < b)) // condition for a

{

middle\_number = a;

return middle\_number;

}

else

{

middle\_number = c;

return middle\_number;

}

}

void test()

{

assert (foo(15, 31, 9) == 15);

assert (foo(32, 24, 12) == 24);

assert (foo(88, 45, 51) == 51);

}

int main()

{

test();

return 0;

}

**Correct Answer** : Pass the correct number of arguments/parameters to the function i.e instead of foo(int a, int b) change it to foo(int a, int b, int c)

**Ex. CBEG69**

**Module: Can you fix it?**

**Binary to decimal conversion.**

The function binarytodecimal( ) will take num as inputs. It should return decimal\_num as output.

For example, if num = 111 , the return value decimal\_num should be 7.

binarytodecimal(111) is returning 8. It should return 7.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int binarytodecimal(int num)

{

int rem,decimal\_num = 0,base = 1;

while ( num > 0)

{

rem = num % 10;

decimal\_num = decimal\_num + rem \* base;

num = num / 10;

base = base \* 2;

}

return base;

}

void test()

{

assert (binarytodecimal(10010) == 18);

assert (binarytodecimal(101) == 5);

assert (binarytodecimal(11001) == 25);

}

int main()

{

test();

return 0;

}

**Correct Answer**: Change the statement return base to return decimal\_num.

**Ex. CBEG70**

**Module: Can you fix it?**

The function power( ) will take base and expo as inputs. It should return the result as output.

For example, if base = 3 and expo = 2 the return value result should be 9.

power(3,2) is returning 27 . It should return 9.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int power(int base, int expo)

{

int result = 1;

while(expo >= 0)

{

result \*= base;

--expo;

}

return result;

}

void test()

{

assert (power(2,3) == 8);

assert (power(5,4) == 625);

assert (power(9,3) == 729);

}

int main()

{

test();

return 0;

}

**Correct Answer**: Change the condition statement while(expo >= 0) to while (expo >0)

**Ex. CBEG71**

**Module: Can you fix it?**

**Time conversion(Hour to min)**

The function HourstoMin( ) will take hr as inputs. It should return min as output.

For example : if hr = 7 then return value min should be 420.

HourstoMin(7) is returning 67 . It should return 420.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int HourstoMin(int hr)

{

int min = hr + 60;

return min;

}

void test()

{

assert (HourstoMin(1) == 60);

assert (HourstoMin(4) == 240);

assert (HourstoMin(9) == 540);

}

int main()

{

test();

return 0;

}

**Correct Answer**: Change the statement min = hr + 60 to min = hr \* 60;

**Ex. CBEG72**

**Module: Can you fix it?**

**Time conversion(Minute to seconds)**

The function MintoSec( ) will take min as inputs. It should return sec as output.

For example : if min = 45 then return value sec should be 2700.

MintoSec(45) is returning 45 . It should return 2700.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int MintoSec(int min)

{

int sec = min \* 60;

return min;

}

void test()

{

assert (MintoSec(25) == 1500);

assert (MintoSec(32) == 1920);

assert (MintoSec(41) == 2460);

}

int main()

{

test();

return 0;

}

**Correct Answer**: Change the statement return min to return sec.

**Ex. CBEG73**

**Module: Can you fix it?**

**Time conversion(Hours to seconds)**

The function HourstoSec( ) will take hr as inputs. It should return sec as output.

For example : if hr = 7 then return value sec should be 25200.

HourstoSec(7) is returning 2520 . It should return 25200.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int HourstoSec(int hr)

{

int sec = hr \* 360;

return sec;

}

void test()

{

assert (HourstoSec(3) == 10800);

assert (HourstoSec(5) == 18000);

assert (HourstoSec(8) == 28800);

}

int main()

{

test();

return 0;

}

**Correct Answer**: Change the statement sec = hr \* 360 to sec = hr \* 3600.

**Ex. CBEG74**

**Module: Can you fix it?**

**Calculate the Absolute value of the variable**

**Ex .** absolute value of 5 is 5 i.e. |5| = 5, whereas

absolute value of -5 is 5 i.e.|-5| is 5.

The function absolute( ) will take num1 as inputs. It should return output as output.

For example, if num1 = -56 , the return value output should be 56.

or if num1 = 92 , the return value output should be 92.

absolute(-56) is returning an error. It should return 56.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int absolute(int num1)

{

int output;

if( num1 > 0)

{

output = num1;

return output;

}

else

{

output = num1 \* --1;

return output;

}

}

void test()

{

assert (absolute(10) == 10);

assert(absolute(-12) == 12);

assert(absolute(-41) == 41);

assert(absolute(89) == 89);

}

int main()

{

test();

return 0;

}

**Correct Answer**: change the statement output = num1 \* --1 to output = num1 \* -1

**Ex. CBEG75**

**Module: Can you fix it?**

**Absolute difference calculator**

**Ex .** if num1 = 12 and num2 = 58 then |num1 - num2| = |12 -58| = |-46| = 46

The function absolute( ) will take num1 and num2 as inputs. It should return output as output.

For example, if num1 = 12 and num2 = 58 , the return value output should be 46.

absolute(12, 58) is returning -12. It should return 46.

Please correct the function to return the right output.

#include<stdio.h>

#include<assert.h>

int absolute(int num1, int num2)

{

int output = num1 - num2;

if(output > 0)

return output;

else

return num1 \* -1;

}

void test()

{

assert (absolute(10 , 56) == 46);

assert(absolute(-12 , -4) == 8);

assert(absolute(41, -31) == 72);

assert(absolute(-89, 12) == 101);

}

int main()

{

test();

return 0;

}

**Correct Answer :** change the statement return num1 \* -1 to return output \* -1