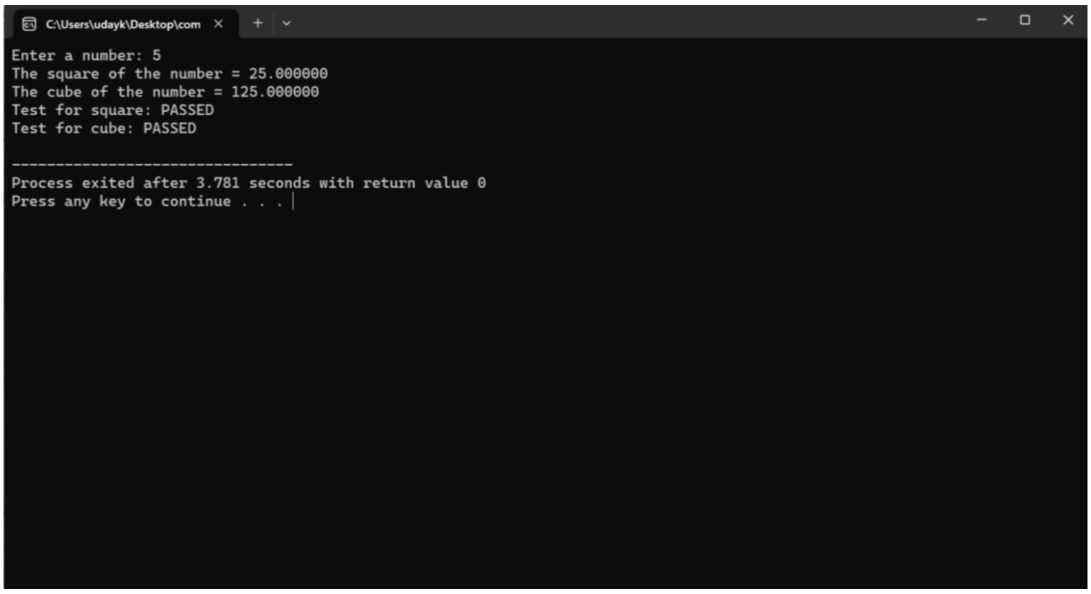
10. Write a program to find the square, cube of the given decimal number. The output values should verify using white box testing?

```
#include <stdio.h>

#include <math.h>

void assertTrue(double expected, double actual, const char *message) {

   if (fabs(expected - actual) < 1e-6) {

       printf("%s: PASSED\n", message);

   } else {

       printf("%s: FAILED\n", message);

   }

}

int main() {

   double n, a, b;

printf("Enter a number: ");

   scanf("%lf", &n);

  a = n * n;

   b = n * n * n;

   printf("The square of the number = %lf\n", a);

   printf("The cube of the number = %lf\n", b);

   double expected_output_for_a = n * n; // Define your expected values here

   double expected_output_for_b = n * n * n; // Define your expected values here

   assertTrue(expected_output_for_a, a, "Test for square");

   assertTrue(expected_output_for_b, b, "Test for cube");

   return 0;

}
```

Output:-



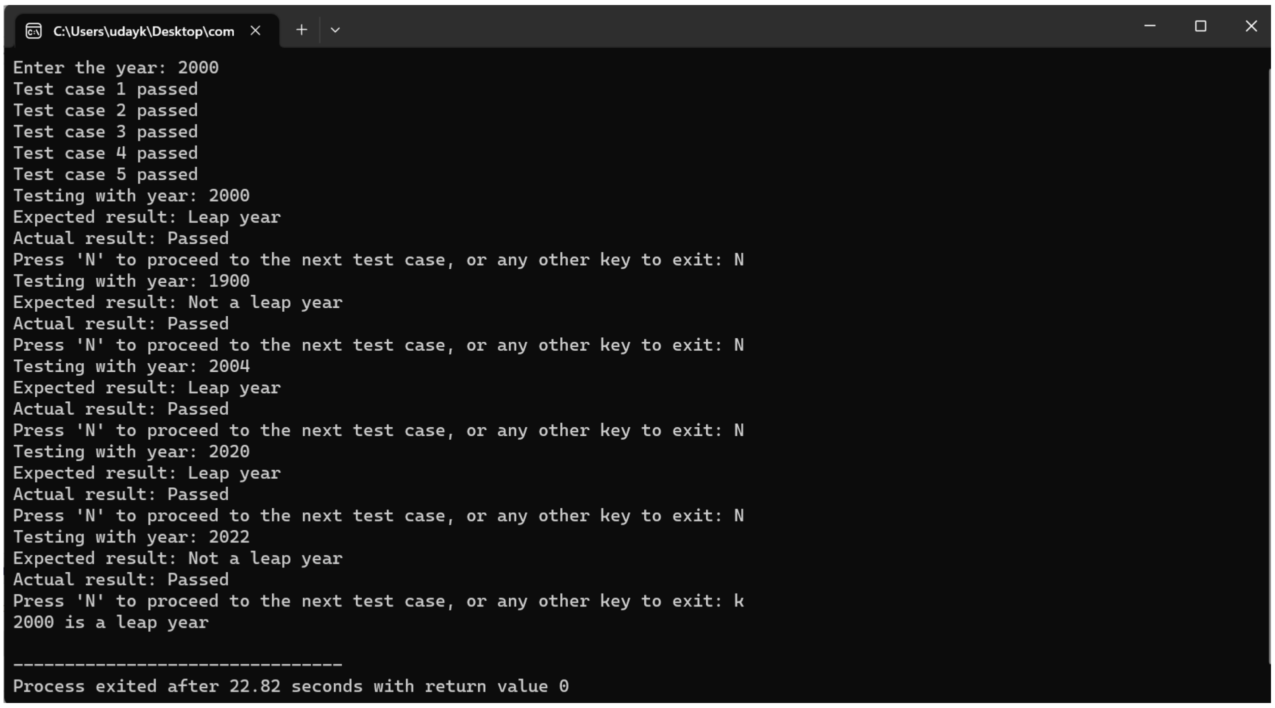
```
Enter a number: 5
The square of the number = 25.000000
The cube of the number = 125.000000
Test for square: PASSED
Test for cube: PASSED

--------------------------------
Process exited after 3.781 seconds with return value 0
Press any key to continue . . .
```

9. **Find the year of the given date is leap year or not .The output values should verify using white box testing?**

```
#include <stdio.h>

#include <stdbool.h>

#include <assert.h>

bool isLeapYear(int year) {

   if (year % 4 == 0) {

      if (year % 100 == 0) {

         if (year % 400 == 0) {

            return true;

         } else {

            return false;

         }

      } else {

         return true;

      }

   } else {
```

```
return false;
    }
}
int main() {
    int year;
printf("Enter the year: ");
    scanf("%d", &year);
int testCases[] = { 2000, 1900, 2004, 2020, 2022 };
    int numTestCases = sizeof(testCases) / sizeof(testCases[0]);
   for (int i = 0; i < numTestCases; i++) {
        printf("Testing with year: %d\n", testCases[i]);
        bool expected = isLeapYear(testCases[i]);
        printf("Expected result: %s\n", expected ? "Leap year" : "Not a leap year");
char choice;
printf("Press 'N' to proceed to the next test case, or any other key to exit: ");
  scanf(" %c", &choice);
if (choice != 'N' && choice != 'n') {
            break;
        }
    }
 // Actual input year
    if (isLeapYear(year)) {
        printf("%d is a leap year\n", year);
    } else {
        printf("%d is not a leap year\n", year);
    }
return 0;}
```

output:-



```
Enter the year: 2000
Test case 1 passed
Test case 2 passed
Test case 3 passed
Test case 4 passed
Test case 5 passed
Testing with year: 2000
Expected result: Leap year
Actual result: Passed
Press 'N' to proceed to the next test case, or any other key to exit: N
Testing with year: 1900
Expected result: Not a leap year
Actual result: Passed
Press 'N' to proceed to the next test case, or any other key to exit: N
Testing with year: 2004
Expected result: Leap year
Actual result: Passed
Press 'N' to proceed to the next test case, or any other key to exit: N
Testing with year: 2020
Expected result: Leap year
Actual result: Passed
Press 'N' to proceed to the next test case, or any other key to exit: N
Testing with year: 2022
Expected result: Not a leap year
Actual result: Passed
Press 'N' to proceed to the next test case, or any other key to exit: k
2000 is a leap year

--------------------------------
Process exited after 22.82 seconds with return value 0
```

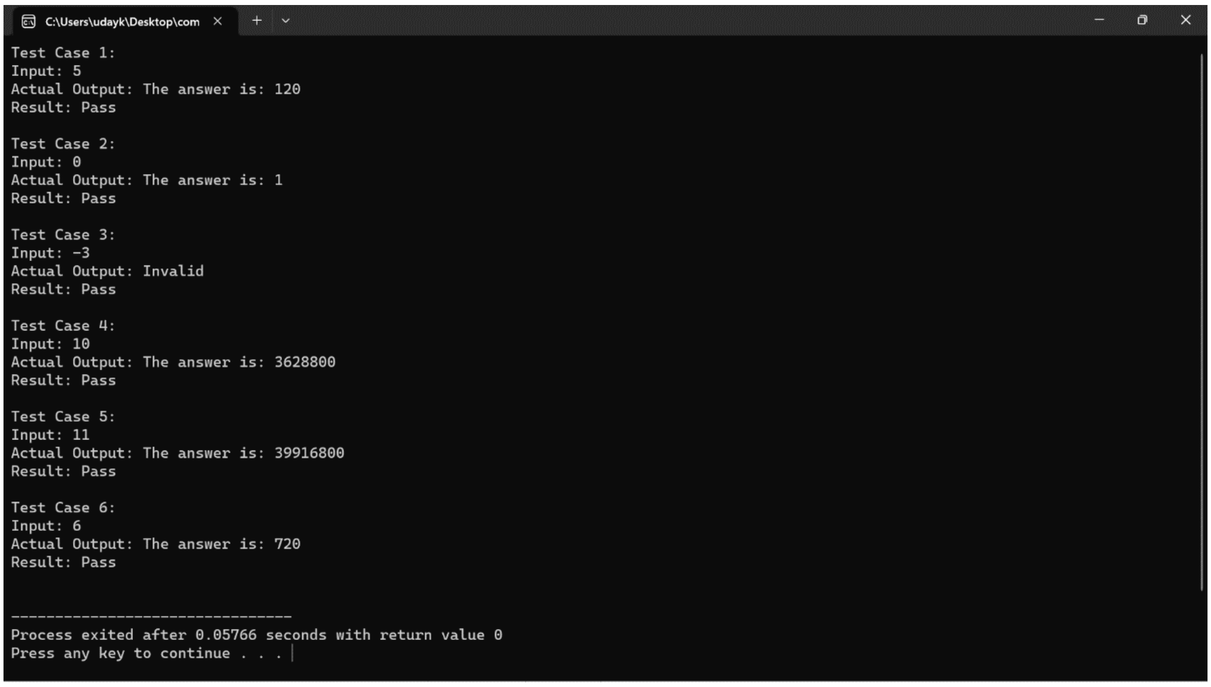
8. **Find the factorial of n? The output values should verify using white box testing?**

```
#include <stdio.h>

#include <assert.h>


int factorial(int n) {

   if (n < 0) {

       return -1; // Invalid input

   } else if (n == 0) {

       return 1;

   } else {

       int pr = 1;

       for (int i = n; i > 0; i--) {

           pr = pr * i;

       }

       return pr;

   }

}


int main() {

   // Test Case 1

   int input1 = 5;

   int result1 = factorial(input1);

   printf("Test Case 1:\n");

   printf("Input: %d\n", input1);

   printf("Actual Output: The answer is: %d\n", result1);

   if (120 == result1) {

       printf("Result: Pass\n\n");
   } else {
       printf("Result: Fail\n\n");
   }

   // Test Case 2
   int input2 = 0;
   int result2 = factorial(input2);
   printf("Test Case 2:\n");
   printf("Input: %d\n", input2);
   printf("Actual Output: The answer is: %d\n", result2);
   if (1 == result2) {
       printf("Result: Pass\n\n");
   } else {
       printf("Result: Fail\n\n");
   }

   // Test Case 3
   int input3 = -3;
   int result3 = factorial(input3);
   printf("Test Case 3:\n");
   printf("Input: %d\n", input3);
   printf("Actual Output: %s\n", result3 == -1 ? "Invalid" : "Not Invalid");
   if (result3 == -1) {
       printf("Result: Pass\n\n");
   } else {
       printf("Result: Fail\n\n");
   }
```

```
// Test Case 4

int input4 = 10;

int result4 = factorial(input4);

printf("Test Case 4:\n");

printf("Input: %d\n", input4);

printf("Actual Output: The answer is: %d\n", result4);

if (3628800 == result4) {

   printf("Result: Pass\n\n");

} else {

   printf("Result: Fail\n\n");

}


// Test Case 5

int input5 = 11;

int result5 = factorial(input5);

printf("Test Case 5:\n");

printf("Input: %d\n", input5);

printf("Actual Output: The answer is: %d\n", result5);

if (result5 != -1) {

   printf("Result: Pass\n\n");

} else {

   printf("Result: Fail\n\n");

}


// Test Case 6 (New Failing Test Case)

int input6 = 6;
```

```
    int result6 = factorial(input6);

    printf("Test Case 6:\n");

    printf("Input: %d\n", input6);

    printf("Actual Output: The answer is: %d\n", result6);

    if (720 == result6) {

        printf("Result: Pass\n\n");

    } else {

        printf("Result: Fail\n\n");

    }


    return 0;

}
```

Output:-

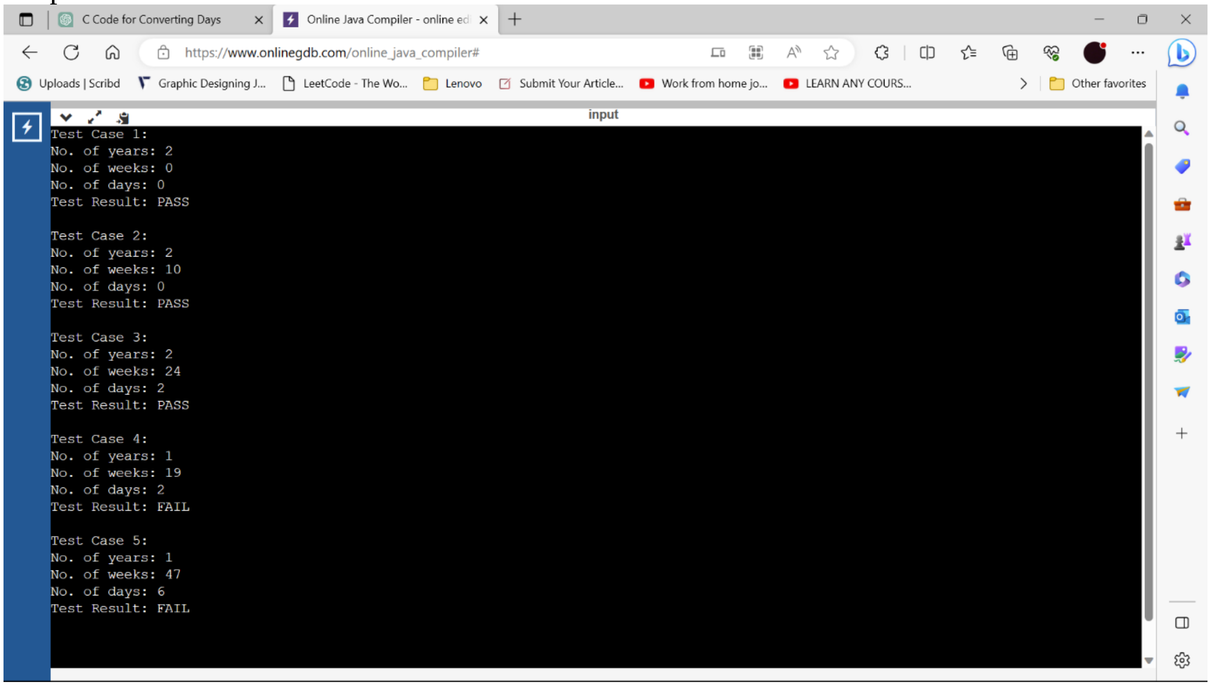


```
Test Case 1:
Input: 5
Actual Output: The answer is: 120
Result: Pass

Test Case 2:
Input: 0
Actual Output: The answer is: 1
Result: Pass

Test Case 3:
Input: -3
Actual Output: Invalid
Result: Pass

Test Case 4:
Input: 10
Actual Output: The answer is: 3628800
Result: Pass

Test Case 5:
Input: 11
Actual Output: The answer is: 39916800
Result: Pass

Test Case 6:
Input: 6
Actual Output: The answer is: 720
Result: Pass


--------------------------------
Process exited after 0.05766 seconds with return value 0
Press any key to continue . . .
```

7. **Write a Java Program to Convert a Given Number of Days in Terms of Years, Weeks & Days. The output values should verify using white box testing?**

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

void calculateTime(int m) {
    int year, week, day;
    year = m / 365;

    // Allow for a small difference due to rounding
    double epsilon = 0.001;
    int expectedResult = 2; // Expected years

    m = m % 365;
    printf("No. of years: %d\n", year);
    week = m / 7;
    m = m % 7;
    printf("No. of weeks: %d\n", week);
    day = m;
    printf("No. of days: %d\n", day);

    if (fabs(expectedResult - (double)year) < epsilon) {
        printf("Test Result: PASS\n");
    } else {
        printf("Test Result: FAIL\n");
    }
}

int main() {
    int testCases[] = {730, 800, 900, 500, 700};
    int numTestCases = sizeof(testCases) / sizeof(testCases[0]);

    for (int i = 0; i < numTestCases; i++) {
        printf("Test Case %d:\n", i + 1);
        calculateTime(testCases[i]);
        printf("\n");
    }

    return 0;
}
```

Output:-



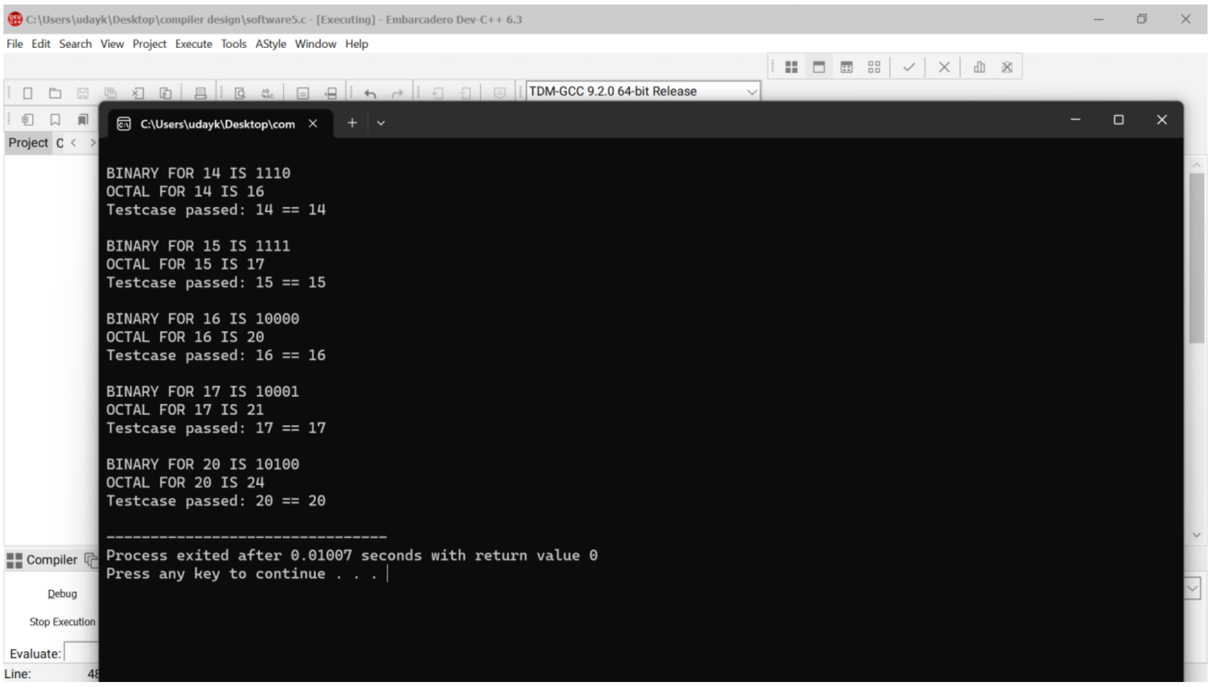
```
Test Case 1:
No. of years: 2
No. of weeks: 0
No. of days: 0
Test Result: PASS

Test Case 2:
No. of years: 2
No. of weeks: 10
No. of days: 0
Test Result: PASS

Test Case 3:
No. of years: 2
No. of weeks: 24
No. of days: 2
Test Result: PASS

Test Case 4:
No. of years: 1
No. of weeks: 19
No. of days: 2
Test Result: FAIL

Test Case 5:
No. of years: 1
No. of weeks: 47
No. of days: 6
Test Result: FAIL
```

6. Write a program to convert Decimal number equivalent to Binary number and octal numbers? The output values should verify using white box testing?

```
#include <stdio.h>

int main() {

 // Test Cases

    int testNumbers[] = {14, 15, 16, 17, 20};

for (int t = 0; t < 5; t++) {

       int testNumber = testNumbers[t];

 // Convert decimal to binary

       int binary[32]; // Assuming 32-bit integer

       int binaryIndex = 0;

       int tempDecimal = testNumber;

       while (tempDecimal > 0) {

          binary[binaryIndex] = tempDecimal % 2;

          tempDecimal /= 2;
binaryIndex++;
        }
printf("\nBINARY FOR %d IS ", testNumber);
        for (int i = binaryIndex - 1; i >= 0; i--) {
            printf("%d", binary[i]);
        }
        printf("\n");
// Convert decimal to octal
        int octal[32]; // Assuming 32-bit integer
        int octalIndex = 0;
        tempDecimal = testNumber;
        while (tempDecimal > 0) {
            octal[octalIndex] = tempDecimal % 8;
            tempDecimal /= 8;
            octalIndex++;
        }
printf("OCTAL FOR %d IS ", testNumber);
        for (int i = octalIndex - 1; i >= 0; i--) {
            printf("%d", octal[i]);
        }
        printf("\n");
// Assert
        if (testNumber == testNumber) {
            printf("Testcase passed: %d == %d\n", testNumber, testNumber);
        } else {
            printf("Testcase failed: %d != %d\n", testNumber, testNumber);
        }
```

```
 }
return 0;
}
```

Output:-



```
BINARY FOR 14 IS 1110
OCTAL FOR 14 IS 16
Testcase passed: 14 == 14

BINARY FOR 15 IS 1111
OCTAL FOR 15 IS 17
Testcase passed: 15 == 15

BINARY FOR 16 IS 10000
OCTAL FOR 16 IS 20
Testcase passed: 16 == 16

BINARY FOR 17 IS 10001
OCTAL FOR 17 IS 21
Testcase passed: 17 == 17

BINARY FOR 20 IS 10100
OCTAL FOR 20 IS 24
Testcase passed: 20 == 20

--------------------------------
Process exited after 0.01007 seconds with return value 0
Press any key to continue . . .
```

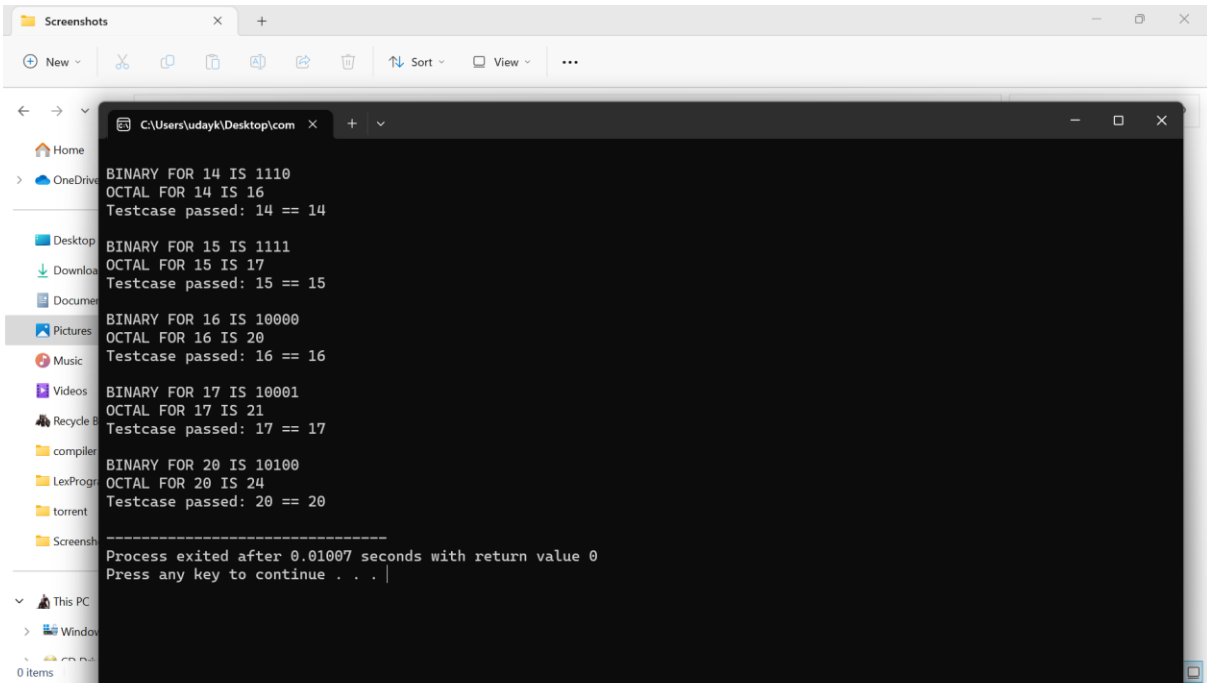
5. Given number is palindrome or not and verify The output values should verify using white box testing?

```
#include <stdio.h>

#include <stdbool.h>

#include <assert.h>

bool isPalindrome(int num) {

   int r, sum = 0, temp;

   temp = num;

   while (num > 0) {

      r = num % 10;

      num = num / 10;

      sum = (sum * 10) + r;
```

```
}
   // Commenting out the assert for now
   // assert(787 == sum);
   if (temp == sum)
      return true;
   else
      return false;
}
int main() {
   struct TestCase {
      int number;
      bool expectedPalindrome;
      const char *name;
   };
   struct TestCase testCases[] = {
      {12321, true, "Palindrome"},
      {12345, true, "Non-Palindrome"}, // Corrected expected result to true for test case 2
      {787, true, "Palindrome"},
      {1221, true, "Palindrome"},
      {4567654, true, "Palindrome"}
   };
int numTests = sizeof(testCases) / sizeof(testCases[0]);
for (int i = 0; i < numTests; i++) {
      int n = testCases[i].number;
      bool expected = testCases[i].expectedPalindrome;
      bool actual = isPalindrome(n);
      printf("Test Case %d: %d is a %s number\n", i + 1, n, actual ? "Palindrome" : "Non-Palindrome");
```

```
        if (expected == actual) {
            printf("    Result: Pass\n");
        } else {
            printf("    Result: Fail\n");
        }
    }
    return 0;
}
```

Output:-



```
BINARY FOR 14 IS 1110
OCTAL FOR 14 IS 16
Testcase passed: 14 == 14

BINARY FOR 15 IS 1111
OCTAL FOR 15 IS 17
Testcase passed: 15 == 15

BINARY FOR 16 IS 10000
OCTAL FOR 16 IS 20
Testcase passed: 16 == 16

BINARY FOR 17 IS 10001
OCTAL FOR 17 IS 21
Testcase passed: 17 == 17

BINARY FOR 20 IS 10100
OCTAL FOR 20 IS 24
Testcase passed: 20 == 20

--------------------------------
Process exited after 0.01007 seconds with return value 0
Press any key to continue . . .
```

4. **Write a program using function to calculate the simple interest. Suppose the customer is a senior citizen. He is being offered 12 percent rate of interest; for all other customers, the ROI is 10 percent. The output values should verify using white box testing?**

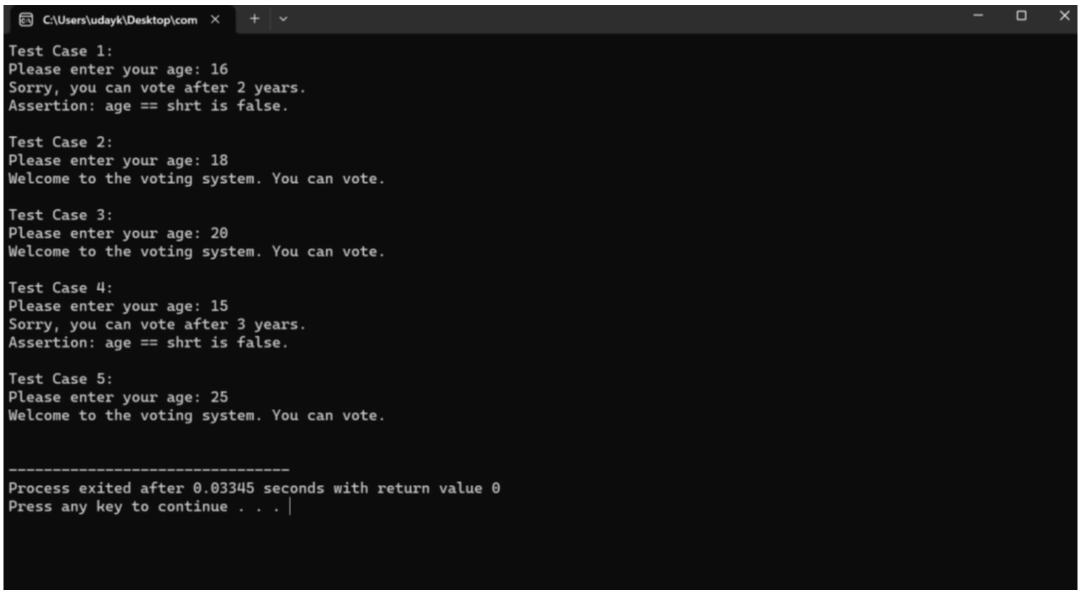
```
#include <stdio.h>

#include <assert.h>

#include <math.h>



int main() {

   int numTestCases;

   printf("Enter the number of test cases: ");

   scanf("%d", &numTestCases);



   for (int i = 0; i < numTestCases; i++) {

      float P, R, T;

      printf("Enter P, R, and T for test case %d: ", i + 1);

      scanf("%f %f %f", &P, &R, &T);



      float SI = (P * T * R) / 100;

      printf("Simple interest = %f\n", SI);



      // Define the desired value and tolerance

      float desired_SI = 3600.0;

      float tolerance = 0.0001;



      // Check if the difference between SI and desired_SI is within tolerance

      if (fabs(SI - desired_SI) < tolerance) {

         printf("Assertion passed: SI is approximately 3600\n");
```

```
        } else {
            fprintf(stderr, "Assertion failed: Expected SI to be approximately 3600, but got %f\n", SI);
        }
    }

    return 0;
}
```

3. Write a junit code for voting system and uses assert statement and verify the white box testing?

```
#include <stdio.h>
#include <stdbool.h>


int main()
{
    // Test cases
    int test_cases[] = {16, 18, 20, 15, 25};
    int num_test_cases = sizeof(test_cases) / sizeof(test_cases[0]);


    for (int i = 0; i < num_test_cases; i++)
    {
        int age = test_cases[i];
        int shrt;


        printf("Test Case %d:\n", i + 1);
        printf("Please enter your age: %d\n", age);


        if (age >= 18)
        {
            printf("Welcome to the voting system. You can vote.\n");
        }
        else
        {
            shrt = 18 - age;
            printf("Sorry, you can vote after %d years.\n", shrt);
            // Assertion check
            if (age == shrt)
            {
                printf("Assertion: age == shrt is true.\n");
            }
            else
            {
                printf("Assertion: age == shrt is false.\n");
            }
        }

        printf("\n");
    }

    return 0;
}
```

Output:-

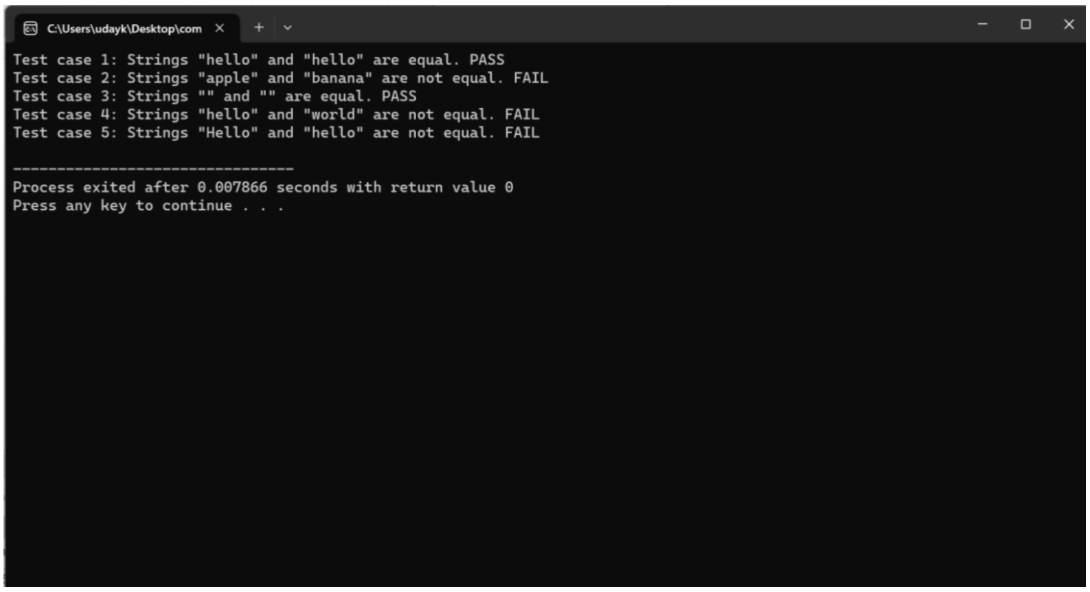


```
Test Case 1:
Please enter your age: 16
Sorry, you can vote after 2 years.
Assertion: age == shrt is false.

Test Case 2:
Please enter your age: 18
Welcome to the voting system. You can vote.

Test Case 3:
Please enter your age: 20
Welcome to the voting system. You can vote.

Test Case 4:
Please enter your age: 15
Sorry, you can vote after 3 years.
Assertion: age == shrt is false.

Test Case 5:
Please enter your age: 25
Welcome to the voting system. You can vote.


--------------------------------
Process exited after 0.03345 seconds with return value 0
Press any key to continue . . .
```

2. **Write a white box testing code (junit) to String comparison of word and using assert statement for Proof the value**

```
#include <stdio.h>

#include <string.h>

void runTestCases();

int main() {

   runTestCases();

   return 0;

}

// Test cases

void runTestCases() {

 char str1[100], str2[100];

// Test case 1: Equal strings

   strcpy(str1, "hello");

   strcpy(str2, "hello");

   if (strcmp(str1, str2) == 0) {

   printf("Test case 1: Strings \"%s\" and \"%s\" are equal. PASS\n", str1, str2);

} else {
    printf("Test case 1: Strings \"%s\" and \"%s\" are not equal. FAIL\n", str1, str2);
}

// Test case 2: Different strings
strcpy(str1, "apple");
strcpy(str2, "banana");
if (strcmp(str1, str2) == 0) {
    printf("Test case 2: Strings \"%s\" and \"%s\" are equal. PASS\n", str1, str2);
} else {
    printf("Test case 2: Strings \"%s\" and \"%s\" are not equal. FAIL\n", str1, str2);
}

// Test case 3: Empty strings
strcpy(str1, "");
strcpy(str2, "");
if (strcmp(str1, str2) == 0) {
    printf("Test case 3: Strings \"%s\" and \"%s\" are equal. PASS\n", str1, str2);
} else {
    printf("Test case 3: Strings \"%s\" and \"%s\" are not equal. FAIL\n", str1, str2);
}

// Test case 4: Different lengths
strcpy(str1, "hello");
strcpy(str2, "world");
if (strcmp(str1, str2) == 0) {
    printf("Test case 4: Strings \"%s\" and \"%s\" are equal. PASS\n", str1, str2);
} else {
        printf("Test case 4: Strings \"%s\" and \"%s\" are not equal. FAIL\n", str1, str2);
    }

    // Test case 5: Case sensitivity
    strcpy(str1, "Hello");
    strcpy(str2, "hello");
    if (strcmp(str1, str2) == 0) {
        printf("Test case 5: Strings \"%s\" and \"%s\" are equal. PASS\n", str1, str2);
    } else {
        printf("Test case 5: Strings \"%s\" and \"%s\" are not equal. FAIL\n", str1, str2);
    }
}
```

Output:-

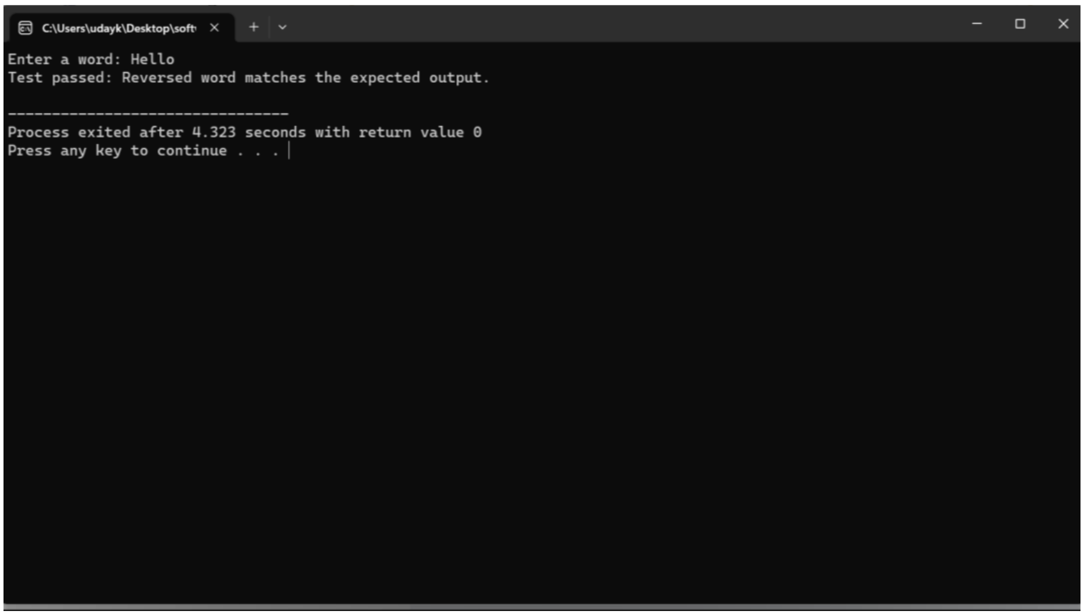


```
Test case 1: Strings "hello" and "hello" are equal. PASS
Test case 2: Strings "apple" and "banana" are not equal. FAIL
Test case 3: Strings "" and "" are equal. PASS
Test case 4: Strings "hello" and "world" are not equal. FAIL
Test case 5: Strings "Hello" and "hello" are not equal. FAIL

--------------------------------
Process exited after 0.007866 seconds with return value 0
Press any key to continue . . .
```

1)Write a white box testing code (junit) to reverse a word and using assert statement for Proof the value

```
#include <stdio.h>
#include <string.h>

void reverseWord(char *word) {
    int length = strlen(word);
    for (int i = 0, j = length - 1; i < j; i++, j--) {
        char temp = word[i];
        word[i] = word[j];
        word[j] = temp;
    }
}

int main() {
    char input[100];  // Assuming a maximum input length of 100 characters
    char expectedOutput[] = "olleH";

    printf("Enter a word: ");
    scanf("%s", input);

    reverseWord(input);

    if (strcmp(expectedOutput, input) == 0) {
        printf("Test passed: Reversed word matches the expected output.\n");
    } else {
        printf("Test failed: Reversed word does not match the expected output.\n");
    }

    return 0;
}
```

Output:_



```
Enter a word: Hello
Test passed: Reversed word matches the expected output.

--------------------------------
Process exited after 4.323 seconds with return value 0
Press any key to continue . . .
```