Week 1 – 2:
--Coding-C-Language Features-Optional.
ROLL NO.:240801153
Name: Kavinkumar S

Status	Finished
Started	Monday, 23 December 2024, 5:33 PM
Completed	Thursday, 17 October 2024, 9:33 AM
Duration	67 days 7 hours

Q1) Write a program to input a name (as a single character) and marks of three tests as m1, m2, and m3 of a student considering all the three marks have been given in integer format.
Now, you need to calculate the average of the given marks and print it along with the name as mentioned in the output format section.
All the test marks are in integers and hence calculate the average in integer as well. That is, you need to print the integer part of the average only and neglect the decimal part.
Input Format :
Line 1 : Name(Single character)
Line 2 : Marks scored in the 3 tests separated by single space.
Output Format:
First line of output prints the name of the student. Second line of the output prints the average mark.
Constraints
Marks for each student lie in the range 0 to 100 (both inclusive)
Sample Input 1 :
A
3 4 6
Sample Output 1 :
A
4

Code:

```c
#include <stdio.h>
int main(){
    char a;
    int m1,m2,m3;
    scanf("%c",&a);
    scanf("%d %d %d",&m1 ,&m2 ,&m3);
    int u= (m1+m2+m3)/3;
    printf("%c",a);
    printf("\n%d",u);
    return 0;
}
```

OUTPUT:

	Input	Expected	Got	
✓	A 3 4 6	A 4	A 4	✓
✓	T 7 3 8	T 6	T 6	✓
✓	R 0 100 99	R 66	R 66	✓

Passed all tests! ✓

Q2) Some C data types, their format specifiers, and their most

common bit widths are as
follows:
· Int ("%d"): 32 Bit integer
· Long ("%ld"): 64 bit integer
· Char ("%c"): Character type
· Float ("%f"): 32 bit real value
· Double ("%lf"): 64 bit real value
Reading
To read a data type, use the following syntax:
scanf("`format_specifier`", &val)
For example, to read a character followed by a double: char ch;
double d;
scanf("%c %lf", &ch, &d);
For the moment, we can ignore the spacing between format
specifiers.
Printing
To print a data type, use the following syntax:
printf("`format_specifier`", val)
For example, to print a character followed by a double: char ch
= 'd';
double d = 234.432;
printf("%c %lf", ch, d);
Note: You can also use cin and cout instead of scanf and
printf; however, if you are taking
a million numbers as input and printing a million lines, it is
faster to use scanf and printf.
Input Format
Input consists of the following space-separated values: int,
long, char, float, and double,
respectively.
Output Format
Print each element on a new line in the same order it was
received as input. Note that the
floating-point value should be correct up to 3 decimal places
and the double to 9 decimal
places.
Sample Input
3

12345678912345
a
334.23
14049.30493
Sample Output
3
12345678912345
a
334.230
14049.304930000

Code:

```c
#include <stdio.h>
int main(){
    int d;
    long s;
    double v,u;
   // float u;
    char c;
    scanf("%d %ld %c %lf %lf ",&d,&s,&c,&u,&v);
    printf("%d \n%ld \n%c \n%.3lf \n%.9lf ",d,s,c,u,v);
    return 0;
}
```

OUTPUT:

	Input	Expected	Got	
✓	3 12345678912345 a 334.23 14049.30493	3 12345678912345 a 334.230 14049.304930000	3 12345678912345 a 334.230 14049.304930000	✓

Passed all tests! ✓

Q3) Write a program to print the ASCII value and the two

adjacent characters of the given
character.

Input Format: Reads the character

Output Format: First line prints the ascii value, second line prints the previous character
and next character of the input character
Sample Input 1:
E
Sample Output 1:
69
D F
Code:

```c
#include <stdio.h>
int main(){
    char c;
    scanf("%c",&c);
    printf("%d",c);

    printf("\n%c %c",c-1,c+1);
    return 0;
}
```

OUTPUT:

	Input	Expected	Got	
✓	E	69 D F	69 D F	✓

Passed all tests! ✓