Simple! Program

- Today we will see some of the simplest C programs.

```
# include <stdio.h>
int main ()  {
   printf("Welcome to ESC101");
   return 0;
}
```

The program prints the message "Welcome to ESC101"

Program components

```
# include <stdio.h>

int main ()
{
    printf("Welcome to ESC101");
    return 0;
}
```

1. This tells the C compiler to include the standard input output library.

2. Include this line routinely as the first line of your C file.

printf is the *function called* to output from a C program. To print a string, enclose it in " " and it gets printed. For now, do not try to print " itself.

"return" returns the control to the caller (program finishes in this case.)

main() is a function. All C programs start by executing from the first statement of the main function.

printf("Welcome to ESC101"); is a statement in C. Statements in C end in semicolon ;

printf

- printf is the “voice” of the C program
  - Used to interact with the users

- printf prints its arguments in a certain format
  - Format provided by user

Another Simple Program

- Program to add two integers (17 and 23).

```
# include <stdio.h>
int main ()  {
   int a = 17;
   int b = 23;
   int c;
   c = a + b;
   printf(“Result is %d”, c);
   return 0;
}
```

The program prints the message: **Result is 40**

```
# include <stdio.h>
int main ()
{
    int a = 17;
    int b = 23;
    int c;
    c = a + b;

    printf("Result is %d",c);
    return 0;
}
```

This tells the compiler to reserve a "box" large enough to hold an integer value. The box is named "a" for use in the rest of the program.

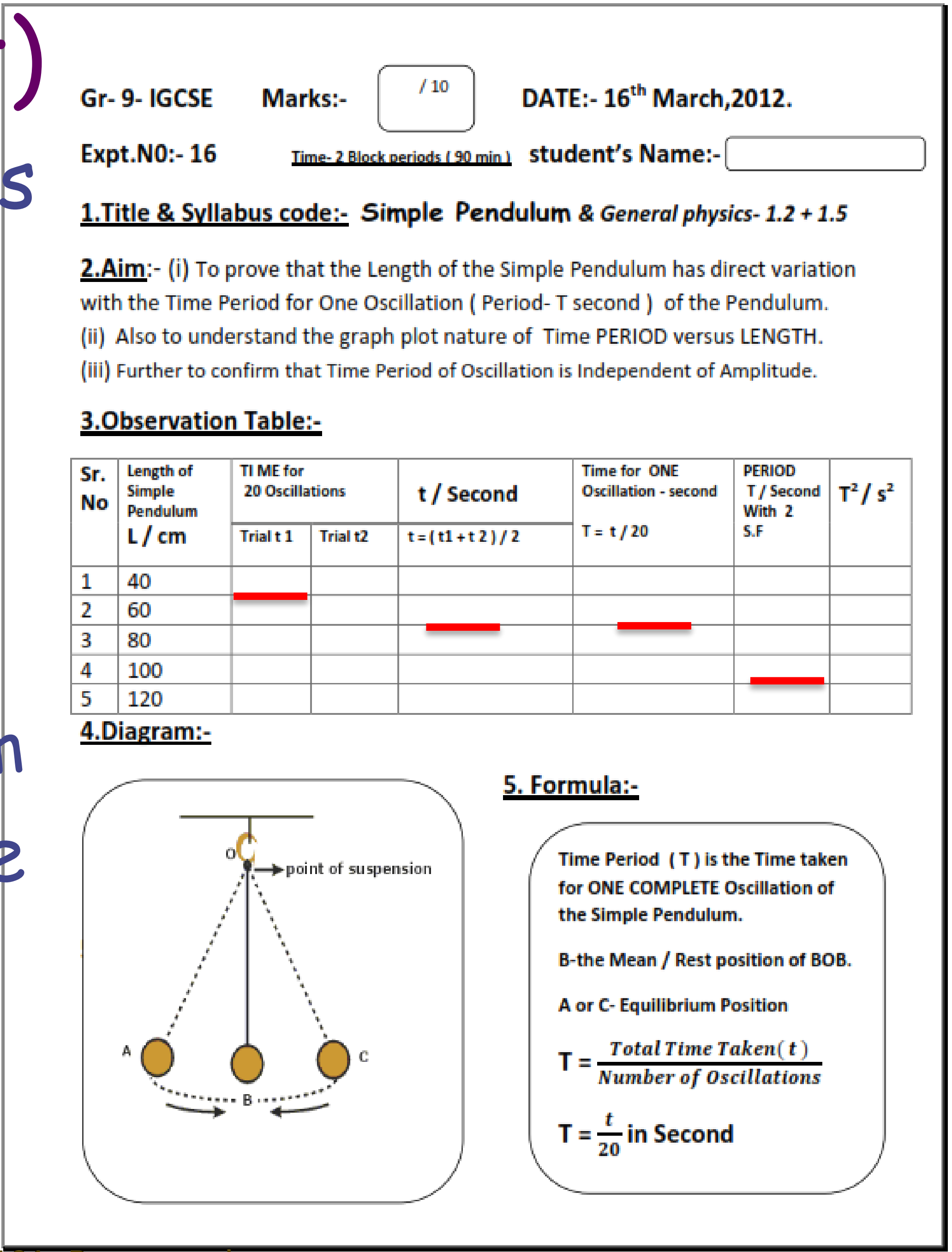
"= 17" stores value 17 in the box that we have named "a". It is OK to skip this part and store value later as we do for box named "c".

+ is an operator used to add two numbers. The numbers come from the values stored in the boxes named "a" and "b"

%d tells printf to expect one integer argument whose value is to be printed. We call it placeholder. We will see more placeholders soon.

printf (% format)

- % format specifiers allow C program to print things whose values are yet not computed
  - will be known while running the program
- %... is similar to the blanks in a lab sheets used for phy/chem labs



Gr- 9- IGCSE Marks:- / 10 DATE:- 16th March,2012.

Expt.N0:- 16 Time- 2 Block periods (90 min) student's Name:-

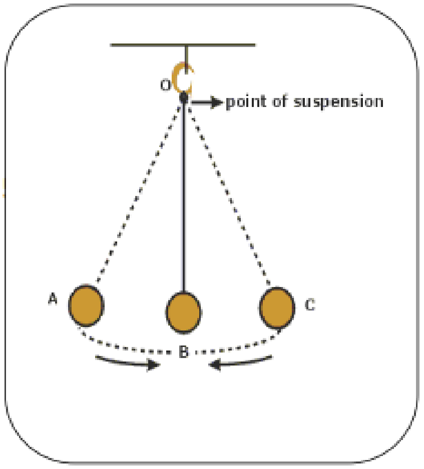
1.Title & Syllabus code:- **Simple Pendulum** & *General physics- 1.2 + 1.5*

2.Aim:- (i) To prove that the Length of the Simple Pendulum has direct variation with the Time Period for One Oscillation (Period- T second) of the Pendulum.
(ii) Also to understand the graph plot nature of Time PERIOD versus LENGTH.
(iii) Further to confirm that Time Period of Oscillation is Independent of Amplitude.

3.Observation Table:-

Sr. No	Length of Simple Pendulum L / cm	TI ME for 20 Oscillations		t / Second	Time for ONE Oscillation - second	PERIOD T / Second With 2 S.F	T^2 / s^2
		Trial t 1	Trial t2	t = (t1 + t 2) / 2	T = t / 20		
1	40						
2	60						
3	80						
4	100						
5	120						

4.Diagram:-



5. Formula:-

Time Period (T) is the Time taken for ONE COMPLETE Oscillation of the Simple Pendulum.

B-the Mean / Rest position of BOB.

A or C- Equilibrium Position

$$T = \frac{Total\ Time\ Taken(\ t\ )}{Number\ of\ Oscillations}$$

$T = \frac{t}{20}$ in Second

Another Simple Program

- A smaller program to add two integers (17 and 23).

```
# include <stdio.h>
int main ()  {
   printf(“Result is %d”, 17+23);
   return 0;
}
```

The program prints the message “Result is 40”

In this case + is operating directly on two integer **constants**.

Types

- **Type:**
  - A set of values
  - A set of operations on these values
- We have been using types
  - Natural numbers
    - 1, 2, 3, … values
    - +, - , *, >, <, … operations
  - Complex numbers
    - 5 + 3i, 7 + 2i, …
    - +, - , *, /, conjugate, …
    - **NO** >, < operations

Data Types in C

- **int**
  - Bounded integers, e.g. 732 or -5
- **float**
  - Real numbers, e.g. 3.14 or 2.0
- **double**
  - Real numbers with more precision
- **char**
  - Single character, e.g. a or C or 6 or $

Notes on Types: char

- Characters are written with ' ' (quotes)
  - 'a', 'A', '6', '$'
- Case sensitive
  - 'a' is not same as 'A'
- Types distinguish similar looking values
  - Integer 6 is not same as character '6'
- Special characters:
  - \n (newline), \' (quote), \" (double quote), \\ (backslash itself), … and many more
  - NOTE: these are SINGLE CHARACTERS, and have to be enclosed in quotes, as '\n'

Escape sequence

More Notes on Types

- Integers (int) are bounded
  - Max value: INT_MAX
  - Min value: INT_MIN
    - These values are system specific
    - -2147483648 … 2147483647 on my machine
- Other data types can only store finite number of values
  - Even some simple real values can not be represented by float and double
- Can surprise you sometimes

```c
#include <limits.h>
#include <stdio.h>
int main() {
    printf("Min=%d, Max=%d",
              INT_MIN, INT_MAX);
    return 0;
}
```

OUTPUT: Min=-2147483648, Max=2147483647

limits.h contains the definitions of INT_MAX and INT_MIN

1. A statement can span multiple lines.
2. printf can use multiple % placeholders.

```
#include <stdio.h>
int main() {
    float y = 100000009.0;
    printf("Value of y is %f", y);
    return 0;
}
```

%f is the placeholder for float.

```
OUTPUT:  Value of y is 100000008.000000
```