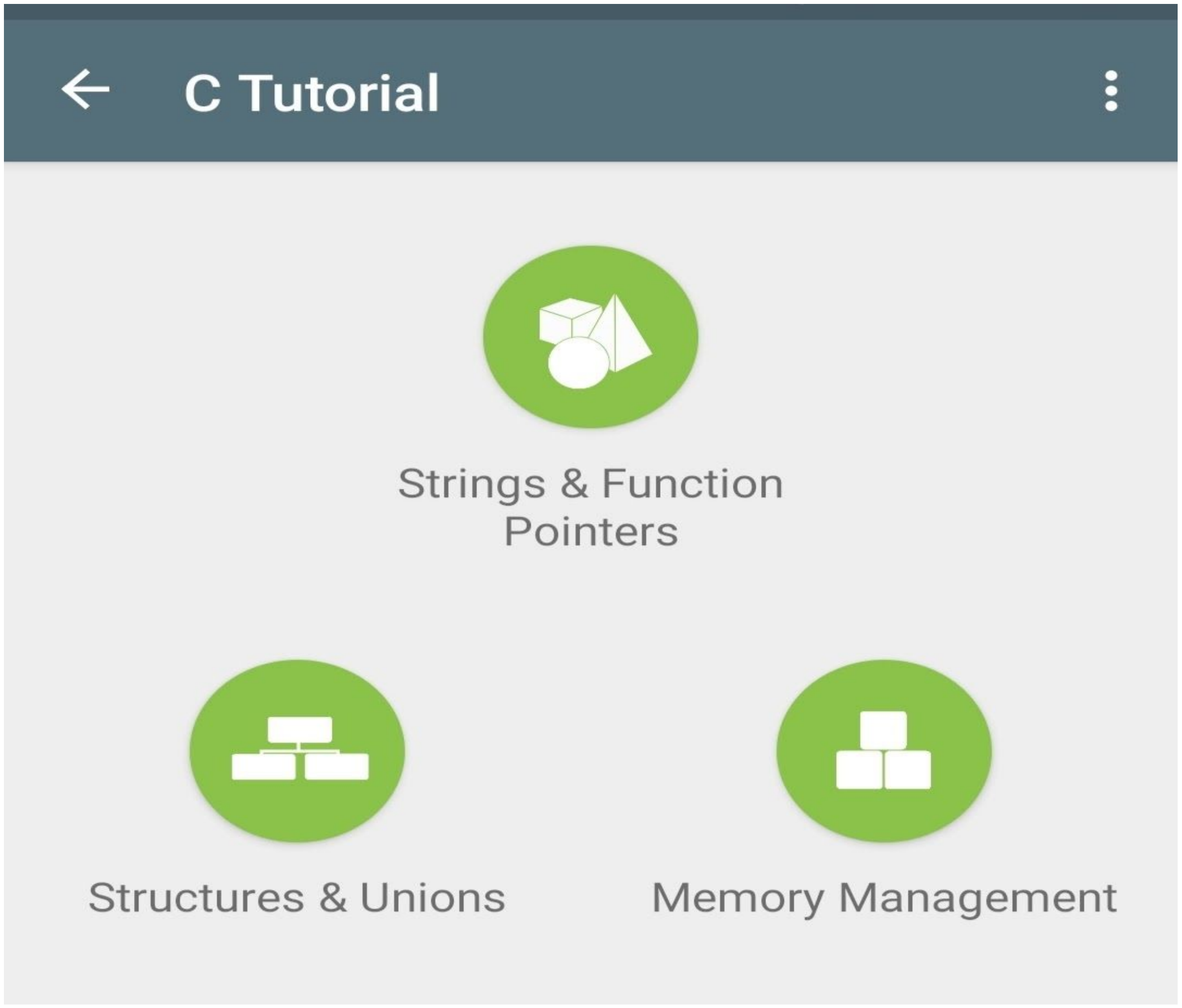
DAILY ASSESSMENT FORMAT

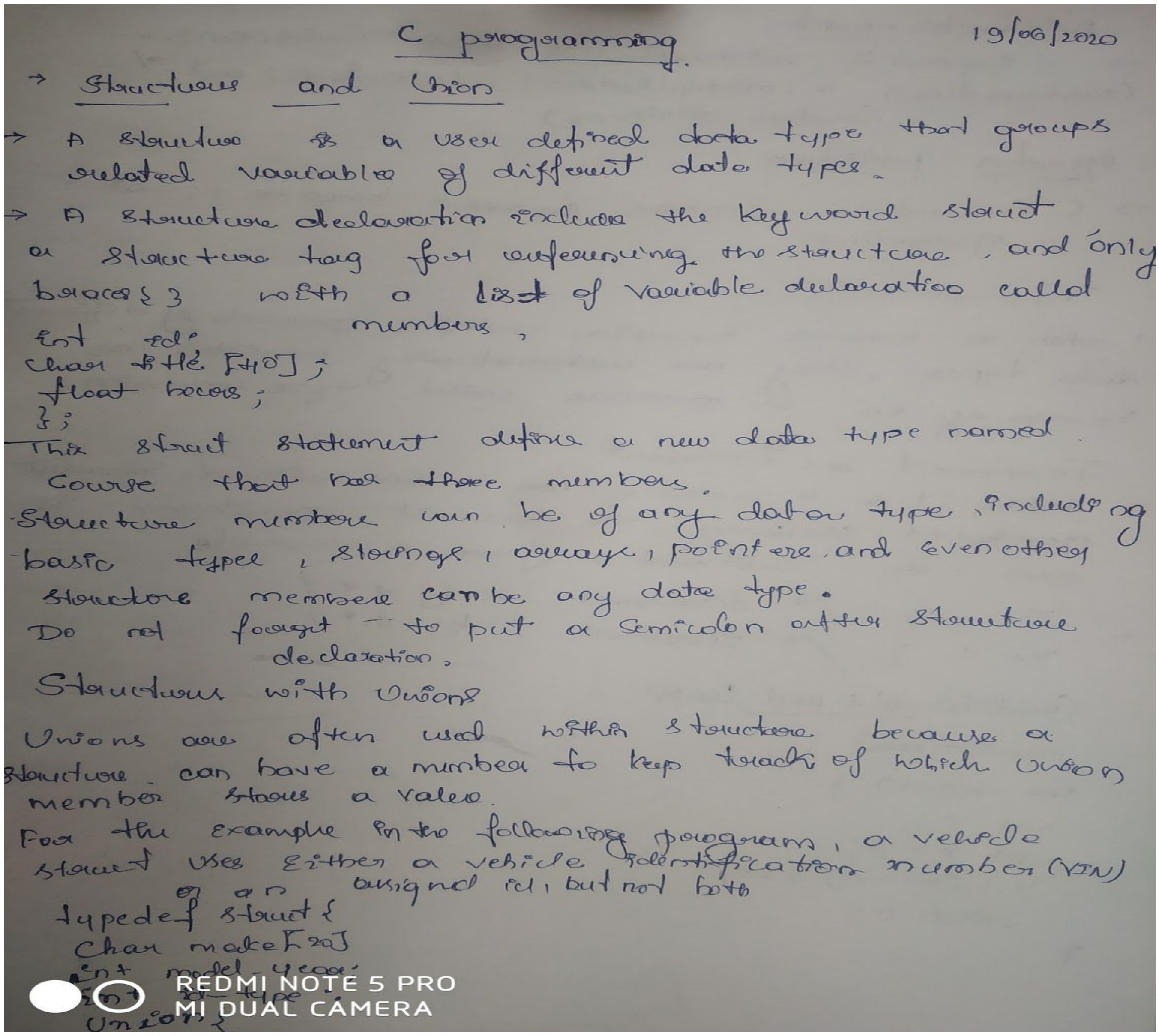
Date:	19-06-2020	Name:	MOUNITHA D M
Course:	C programming	USN:	4AL17EC055
Topic:	Structures and unions Memory Management	Semester & Section:	6TH SEM "A" SEC
Github Repository:	Mounitha_-ec055		

FORENOON SESSION DETAILS

Image of session



Report – Report can be typed or hand written for up to two pages.



C programming 19/06/2020

→ Structures and Union

→ A structure is a user defined data type that groups related variables of different data types.

→ A structure declaration includes the keyword struct a structure tag for referencing the structure, and only braces { } with a list of variable declaration called members,

```
int id;
char title [40];
float hours;
};
```

This struct statement define a new data type named Course that has three members.

Structure members can be of any data type, including basic types, strings, arrays, pointers, and even other structure members can be any data type.

Do not forget to put a semicolon after structure declaration.

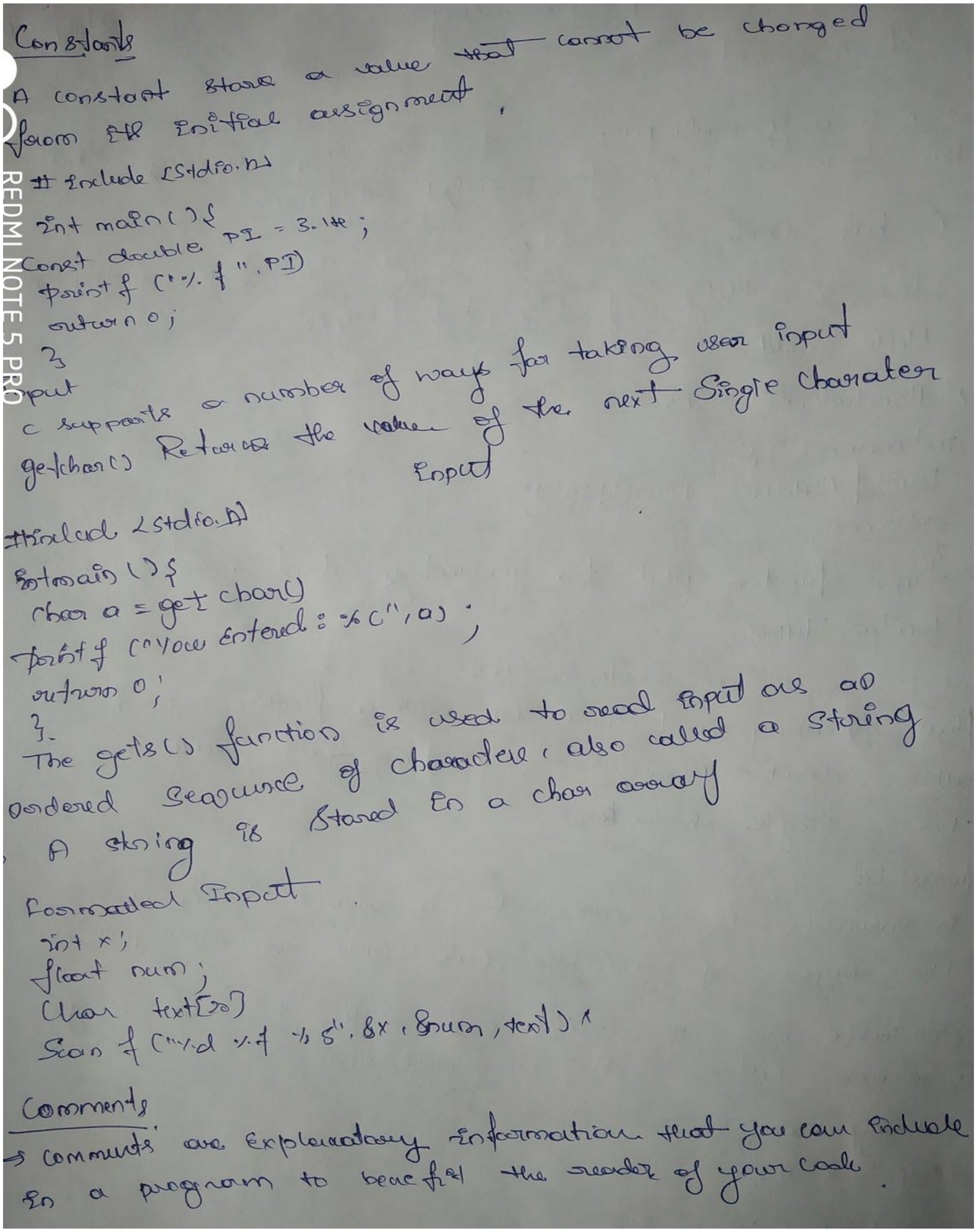
Structures with Unions

Unions are often used within structure because a structure can have a member to keep track of which union member stores a value.

For the example in the following program, a vehicle struct uses either a vehicle identification number (VIN) or an assigned id, but not both

```
typedef struct {
 char make[20]
 int model-year;
 int id-type;
 union {
```

REDMI NOTE 5 PRO
MI DUAL CAMERA



Constants

A constant store a value ~~that~~ cannot be changed from the initial assignment.

```
# include <stdio.h>
int main(){
Const double PI = 3.14 ;
printf ("%f", PI)
return 0;
}
```

Input

C supports a number of ways for taking user input

getchar() Returns the value of the next single charater input

```
#include <stdio.h>
int main(){
char a = get char()
printf ("You entered: %c", a) ;
return 0;
}.
```

The gets() function is used to read input as an ordered sequence of character, also called a string

A string is stored in a char array

Formatted Input

```
int x;
float num;
Char text[20]
Scan f ("%d %f %s", &x, &num, text) ^
```

Comments

→ comments are Explanatory information that you can include in a program to benefit the reader of your code.

Arithmetic operators

→ C supports arithmetic operators +(addition), - (subtraction), * (multiplication), / (division) and % (modulus division)

Operator precedence

→ C Evaluates a numberic expression based on operator precedence

Type conversion

→ when a numeric expression contains operands of different data types, they are automatically converted as necessary in a process called type conversion,

Increment and Decrement

→ Adding 1 to a variable can be done with the increment operator++, similarly, the decrement operator -- is used to subtract 1 from a variable

```
z--;  /* decrement z by 1*/
y++;  /* increment y by 1*/
```

Conditionals and loops

→ conditional

```
int main() {
  int y;
  int x = 3;
  y = (x>=5) 5:x;
   return 0;
}
```