ICP

1ST SEMESTER

ABSTRACT

ICP is very abstract level and helpful for computer science students. All basic programing rules and regulation are define and code in C++ language is use to explain the example I dedicate this book to myself because becoming an Engineer is not easy task. Special thanks to my parents and teacher.

Junaid Saturday, 09

INTRODUCTION TO COMPUTER PROGRAMMING

Copyright

AUTHOR

Junaid

Software Engineer

COMSATS University, SAHIWAL

00-92-345-7448497

https://www.facebook.com/junaidbilal005

ghauricenter@gmail.com

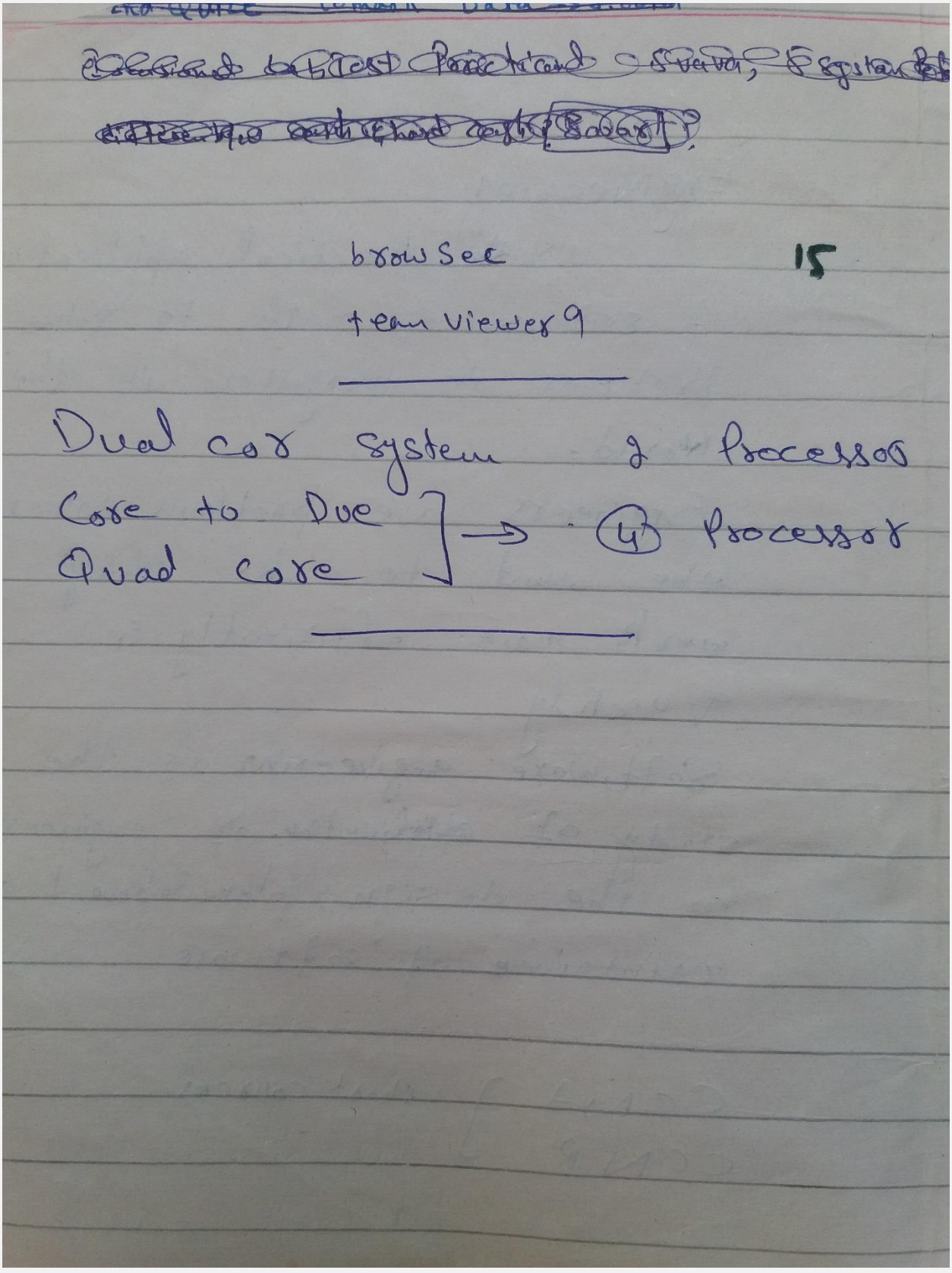
Comsats Sahiwal. 14

Engineering is a practical application of science & math to solve problems in everywhere in the world.

Engineers are problems solvers who want to make things work more efficiently & quickly.

Software engineering is the study of application of engineering to the design, development & maintaince of software.

CCNA
CCNP
} short courses.



browSec

15

team Viewer 9

Dual cor system → 2 Processor

Core to Duo

Quad core → 4 Processor

Collection of Program is Software. 16

Computer is an electronic device that process data into meaningful information

Data :- Raw fact & figures are called data.

Information is meaning ful form of data. is called information.

Integrated circuit (IC) :- An integrated circuit consists of thousands of transistors & other electronic components on a single crystal.

Input & output / input device & output device.

set Refers to a collection of related entries.

Programing :- A set of instructions that tells the computer what to do

Algorithm :- Step by step to solve a Problem.

$A = \pi r^2$ & $C = 2\pi r$

17

1- Start.
2- input A] input / Data
3- Input B]
4- Sum = A+B] Processing
5- Display sum] output / information.
6- End.

Logic Design — وہ ترتیب جس کو ہم ڈیزائن کرتے ہیں

Problem statement. means to find & explain them — کسی بھی Problem کو پہلے ڈھونڈتے ہیں Pro. stat ہے۔

1- System Unit. consist of different component that work together on data according to instructions. (RAM, CPU)
2- Mother Board or System Board.
3- RAM → Radom Access memory, Temp, voltial, Direct & Main میموری Primary memory, R/W
4- ROM → Permanent memory, Hard disck.
non-voltial, Read only memory.

Types of RAM
SRAM → static RAM
DRAM → Dynamic RAM
Page # 40
→ D-RAM's are cheapies & lower

Types of ROM
PROM
EPROM
EEPROM (Electrically Erasable Programmable ROM)
Page # 41

18

⇒ TFT (LCD) thin Film Transistor.

⇒ Five Phase to develop a Software.

① Logic Design ② Algorithm ③ cording
④ Testing ⑤ Final Documentation.

⇒ 1- Start
2- Input r
3- $A = \pi r^2$
4- $C = 2\pi r$
5- Display A
6- Display C
7- End

1- Start
2- Input r
3- $C = 2\pi r$
4- Display C
5- End.

1- Start
2- Input A
Input B
3- Input C
4- $D = A * B$
5- $E = D/C$
6-
7- Disply D
8- Disply E
9- End

⇒ اختیار کیا Pro. statement اس کو بنا کر دینا information کہتے ہیں Problem Statement

⇒ C Language → class → objects.
C++ Lasted virsion.

⇒ 1 bit — Binary digit (0 and 1)

8 bits	1 Byte
1024 bites	1 Kb
1024 Kb	1 Mb
1024 Mb	1 Gb
1024 Gb	1 Tb
1024 Tb	1 Petab
1024 Pb	1 Exab
1024 Eb	1 Zetta b
1024 Zb	1 Yota b
1024 Yb	1 Bronto b

(Page # 63)

PEZY | B

$a \propto b$ & $a \propto \frac{1}{a}$ 19

Problems Statements & No. of steps in Algorithm

=> Serial ports & Parallel Ports (Page # 46)

=> Main parts of CPU (or) unit.
ALU & C.U (Page #(35)

=> Register. (Page # 38)

=> cash Memory (Page # 42)

=> No. System & Types of Data.

Types of Data.

① Numeric Data :- consists of Numeric digits from 0 to 9 - it may be +ve or -ve digits. 10, -12, -32.5 etc
it may also contain decimal point.

② Alphabetic Data.
it consists of Alphabetic from A to Z in both cases uper case & Lower case. It also contain Blank space.
A -> Z & a -> z + M USMAN ← space.

20

③ Alphanumeric Data.
consits of special symbols & combination of Alphabetic & Numeric data.

④ Image Data. → consits of chart, Graphs, Pics
⑤ Audia. Data. Sound is a representation of audio. Audio data consits of music, speech. or any type of audia.
⑥ Video:- Video is a set of full-motion images played at high speed. Video is used to display actions & movements. → 3D

Book #38

No. System.

① Binary No. Sys (0,1) Base 2
② Octal No. Sys (0,7) " 8
③ Decimal No. Sys (0,9) " 10
④ Hexadecimal No. Sys. (0-9, A, B, C, D, E, F) " 16

=> Frount End & Back End.

Coding Schemes which show the Binary Code.

Data Representation

Coding Scheem 21

① BCD code. (Binary coded Decimal) code

⇒ it is 4 bit binary code.

⇒ Each Decimals digits represents 4 bit binary code.

⇒ Use in early computers.

0000 → 0 0011 → 3 0110 → 6

0001 → 1 0100 → 4 0111 → 7

0010 → 2 0101 → 5 1000 → 8

1001 → 9.

8 bits = 1 byte (256 characters.)

② ASCII code (American standard code for Information Interchange) code

→ Published by ANSI (American National Standard Institute). 1968

→ It is 7 bit binary code.

→ It is represents 128 characters.

→ Use in Personal Computers.

0 - 31 Blank space.

32 - Space bar

33 - 64 Blank space.

65 - 90 A → Z

91 - 96 Blank space

97 - 122 (a → z)
123 - 126 Back Space
127 Delete.

22

ALI
65 76 73 → 1011101.
100100
1000001

③ EBCDIC → (Extended Binary coded Decimal Interchange code)
(0 - 255)

→ Replace by ASCII
→ It is 8 bit binary code.
→ It is Represents 256 characters.
→ use in Mainframe Computers.

④ Uni Code.
→ It is 16 bit binary code.
→ Represents (256 * 256) or 65536 characters.
→ use in All world Languages.

✓ 23

Program Development Process. (PDP)

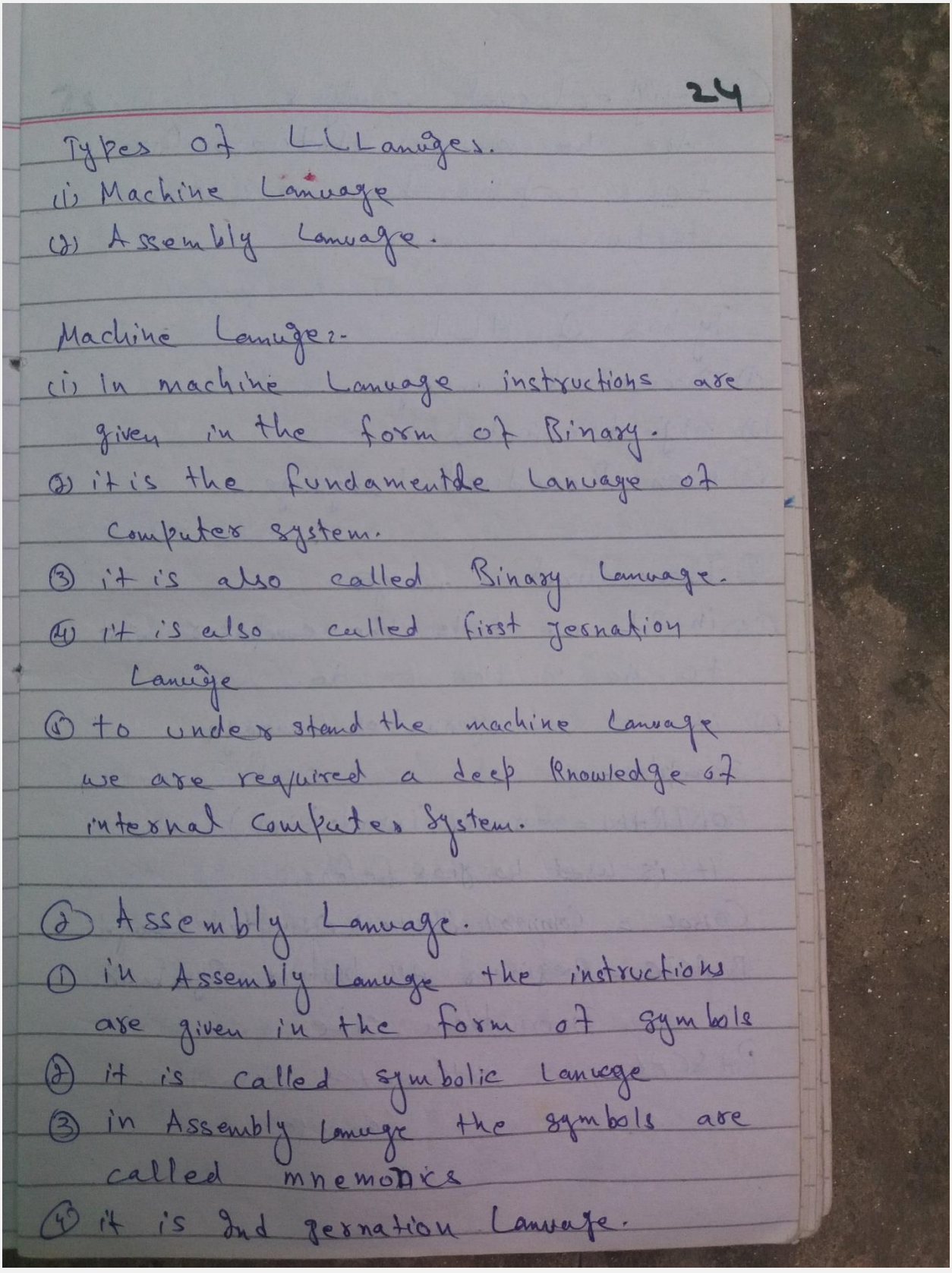
(1) Analyzing & Defining the problems.
(2) Designing an Algorithm.
(3) Coding (OR) writing the program.
(4) Testing
(5) Final Documentation.

=> Programming Languages.

A sets of words, symbols & codes use to write a programs is called programming Lanuage.
Two types of Lanuage.
(1) Low level Lanuage.
(2) Higher level Lanuage.

Page #95

(1) Low Level Lanuge: LLL are the languages that are close to Computer hard ware & far from Human language.



24

Types of LL Languages.

(1) Machine Language
(2) Assembly Language.

Machine Language:-

(1) In machine Language instructions are given in the form of Binary.
(2) it is the fundamentle Language of Computer system.
(3) it is also called Binary Language.
(4) it is also called first gernation Lanuge
(5) to understand the machine Language we are required a deep Knowledge of internal Computer System.

(2) Assembly Language.

(1) in Assembly Lanuge the instructions are given in the form of symbols
(2) it is called symbolic Lanuage
(3) in Assembly Lanuge the symbols are called mnemonics
(4) it is 2nd gernation Lanuage.

(2) Higher Level Languages.

Are the languages that are far from computer hardware & close to Human Lanuge.

Types of HLL.

1. Procedural Lanuage.
2. object oriented "
3. Non-Procedural Language.

(1) Procedural Lanuage.

(1) => in P.L we tells the computer what to do & How to do.

(2) it is 3rd gernation Lanuage

Examples.

FORTRAN = Formula Translation)
it is used 40 year before.

COBOL = Common Business Oriented Lanuge

BASIC = Beginner All purpose Symbolic Intruction code.

PASCAL = is Mathematics Saiencetics. Lanuege.

(2) Object Oriented Lanuage.

in OOL we write Programs on the basic of object.

Object is a Entity that represents Person, Place, or any things.

class
↳ objects.
↳ Properties
→ funtions.

Example · C++, JAVA (GUI) Graphical user interfarance window.

Back End → C++

Frount End → JAVA.

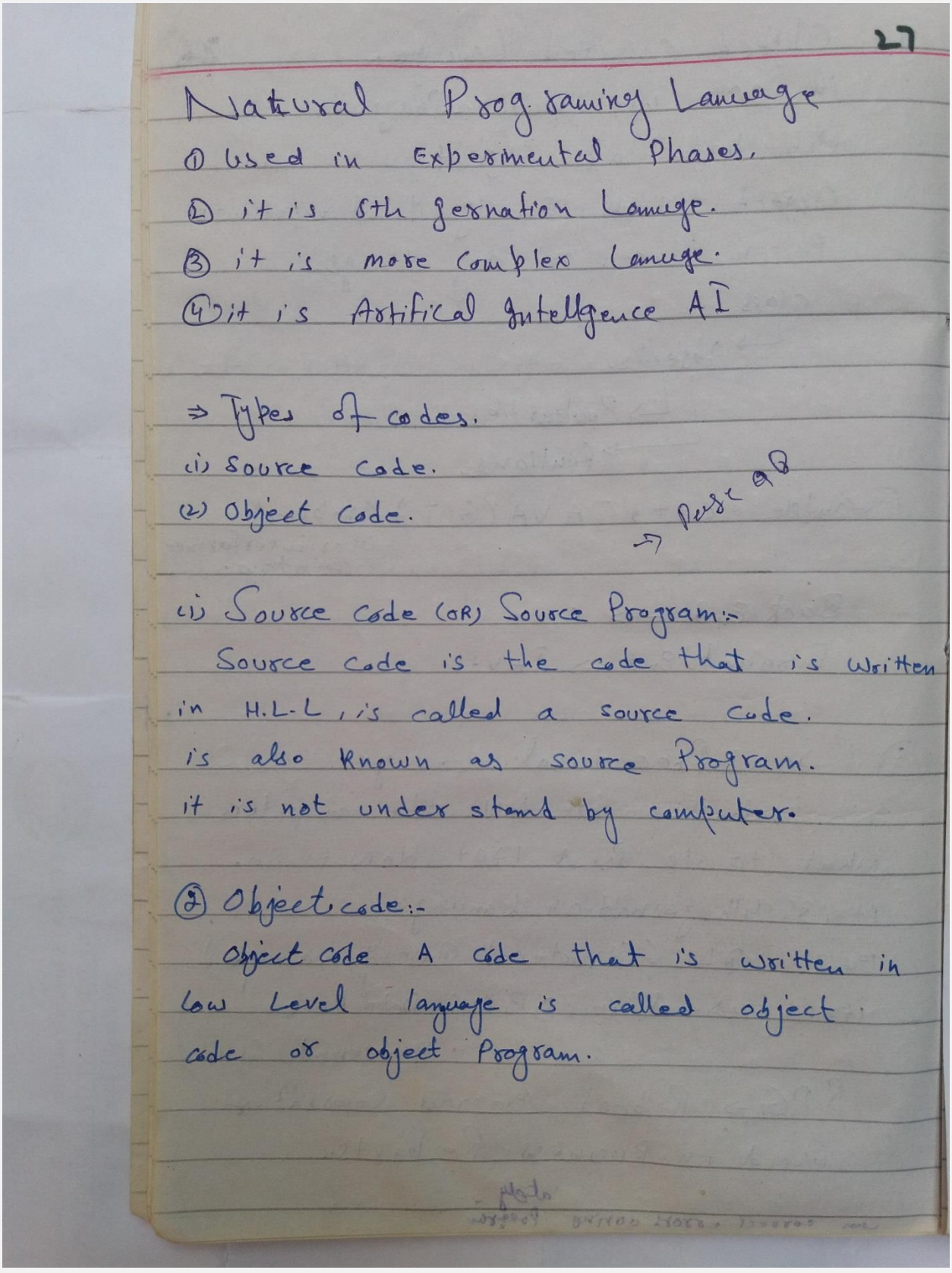
(3) Non-Procedural. Lanuage.

in NPL we tells the compators what to do But Not How to do.

it is 4th geonation Lanuage.

Ex. SQL (Structured Query Lanuage) it is used in DBMS (Data Base management System)

RPG → Reboot Program Geonater used in Business Reports.



27

Natural Programing Lanuage

① Used in Experimental Phases.
② it is 5th gernation Lanuge.
③ it is more complex Lanuge.
④ it is Artifical Intellgence AI

⇒ Types of codes.

(1) Source code.
(2) Object code.

(1) Source code (OR) Source Program:-

Source code is the code that is written in H.L.L, is called a source code. is also known as source Program. it is not under stand by computer.

(2) Object code:-

Object code A code that is written in low Level language is called object code or object Program.

Language Processor or Translater: 28

Translaters is the software that converts the instructions of the other languages into machine language Each computer language has its own translaters.

⇒ Types of Translater.

(i) Compiler
(ii) Assembler
(iii) Interpreter.

→ Page 98

(i) Compiler converts the instructions of H.L.L into machine language.

H.L.Ls → [compiler] → machine language.

(2) Assembler:- Converts the instruction of Assembly language into machine language.

Assembly Language [Assembler] → Machine language.

mnemonics are converted

(3) Interpreter:- Converts the instruction of H.L.L into machine language.

H.L.L [interpreter] → Machine Language.

The Advantage of interpreter over compilers is that errors is found immediately. So the programer can correct errors during Program development.

29

=> Hard ware :- Physical components of Computer is Hard ware,

=> Soft ware :- Program is also called software.

=> Types of software → Page: 87

(1) System Software

- Operating System
- Utity Program
- Device Drivers.

System Software is the collection of programs that is used to manage & Control the actual operation of computer Hardware.

(i) Operating System :- is the collection of Programs that is used to manage the computer components window, Dos, Unix Linux,
(C, D, E) drives
C is operating sys

Application Software.

=> it is include when an application is installed on our computer. also known Application Package.

Data base software :
- (oracle)
- (MS Access)

sbread sheet software
- MS Excel

(2) Graphic software
- coreldraw & Adobe photoshop.

Ulity Programs :- are use for specific task. it is also used to solve the common problems of Hardware & software.

antivirus is Ulity Program

(3)

30

Device Driver:- is used when a device is attach to computer system.

Ex- Printer, Scaner.

⇒ Flow chart.

Graphically representation of Algorithm is called flow chart.

Symbols.

1 oval — oval for start / End.

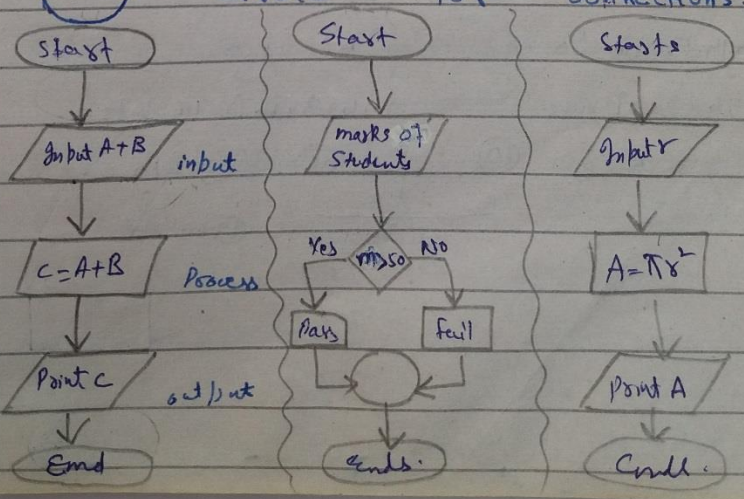
2 Parallelogram — Parallelogram for input / output.

3 Rectangle — Rectangle for Process.

4 Diamond — Diamond for selection / condition.

5 — flow lines for Direction.

6 — circle for connections.



Four Types of Data types.

31

⇒ Data Types is for taking input

1 bytes = 8 bits. = 256 characters.

1- Integer data type:-

(a) int take 4 bytes → (+ve, -ve)

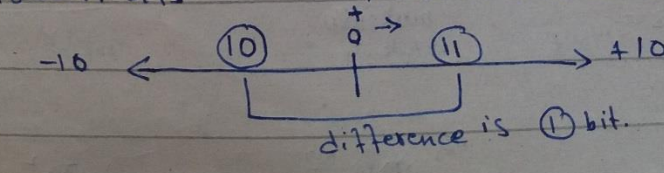
(b) Short // 2 // → (+ve, -ve)

(c) long // 4 // → (+ve, -ve)

(d) Unsigned int // 4 // → (only +ve)

(e) Unsigned long // 4 // → (only +ve)

Signed D. Type (+ve, -ve)	Un-Signed D-Type (only +ve)
1 byte $= 2^7 = 128$ bits	1 byte $= 2^8 = 256$ bits
2 byte $= 2^{15} = 32768$ bits	2 byte $= 2^{16} = 65536$ bits
3 byte $= 2^{23} = 8388608$ bits	3 byte $= 2^{24} = 16777216$ bits
4 byte $= 2^{31} = 2147483648$ bits	4 byte $= 2^{32} = 4294967296$ bits
Range	Range
+2147483648 to	0 to
-2147483647 bits.	+4294967296 bits



difference is (1) bit.

32

(2) Real Data Type.

(a) Float → 4 bytes (+ve, -ve)
(b) double → 8 bytes (+ve, -ve)
(c) long // → 10 bytes (+ve, -ve)

value in points

Signed data type.

(3) Character data type.

1 byte → char

Use by ASCII coding scheem

follow char data type.

1 Byte → 2^7 → 128 (charters) → 0 – 127

(4) Boolean Data type.

1 bits → bool (0,1) use for comparision

true / fasle

Bool is Signed Data Type.

Variable or Indentifier:-

variable or Indentifier is the named memory location (or) cell.

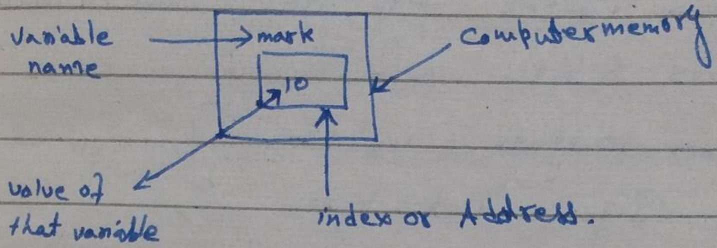
For example → Input A

Input B

A & B are two variables or Identifier

33

& the purpose is to store the value.



=> Types of variables

(i) Standard variable

(ا) اس کا مطلب ہے کوئی بھی ہو سکتا ہے تو
int کا مطلب ہے ہو سکتا integer (±ve)

Input A → Cin → for input
Input B → cout → for output

(2) User defined variables.

User defined variables are those variables which are defined by user. eg → A, B, C, age etc.

=> Key word (or) Reserve word.

(i) Predefined set of instruction
(2) Built in-Key words.

Bool, cout, cin C++ نے اسے، ویسے define کیے ہیں
char اور یہ Built in ہے۔

e.g. Total No of key words is 32

auto, return, signed
break, short, unsigned
case, else, static
char, do, while
for, double, void
if, int, union, long

Keywords is a word in C++ Lanuage that has a predefined meaning & Purpose. The purpose & meaning in define by the developer of Lanuge. it cannot be changed or refine by the user.

جب user سے ہم value لیتے

34

Declaring the variable. کسی چیز کو بیان کرنا

variable کو C++ میں کس طرح لکھ سکتے ہیں۔

Syntax:-

Syntax is the rule which help to understanding These rules govern the way in which instruction of programs structured. Statements will not be executed if they are not correctly written/formulated.

Data type	space	variable name ;
int	space	marks;
int	space	marks, age;
Float	Percentag;	
Char	grade;	
bool	A;	

Data Type کو ہمیشہ small cap میں لکھا جاتا ہے۔

=> Rule for writing variables.

① variable جو بھی ہو اس کا پہلا لفظ (A->Z / a->z) Charcter
! system - کا نام میں ہو - 9mint X, 2KgX, @liv, #AX
② variable کے درمیان space نہیں ہونا چاہیے۔
③ variables کو دونوں small/capital / caps میں لکھ سکتے ہیں۔

35

④ یعنی Keywords ہے ان کو variables کے طور پر استعمال نہیں کر سکتے۔

⑤ یعنی Special Symbol ہے ان کو #, *, >, < variables کے طور پر استعمال نہیں کر سکتے۔

=> Initializing the variable {اس میں variable کو value دیتے ہیں}
(جب user سے value نہیں لیتے)

Syntax:-

Data type space variable name = value of variable;
Assignment operaters

int space marks = 100;
int space marks = 10, age = 20;
float space per = 55.5;
char space grade = 'F';
bool space A = false;

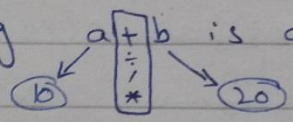
Keywords کی تعداد C++ میں 63 ہے۔

order of Data types.
long double → Highest
double
float
long
int
short
char → Lowest.

Expression:- Returns a single value. 36

→ Combination of operators & operands.

→ eg a + b is an expression.



+, −, /, * are operaters.

a & b are operands.

10 + 20 = 30 it return a single value.

Operands are also called variables.

Operators

There are two types of operators.

1- Uniary operators (one operand)

2- Binary operators (two Operand)

Examples:-

Uniary operators.

a++, b−−, ++b, a+, a−

Binary operantors

a+b, a−b, a*, a/b

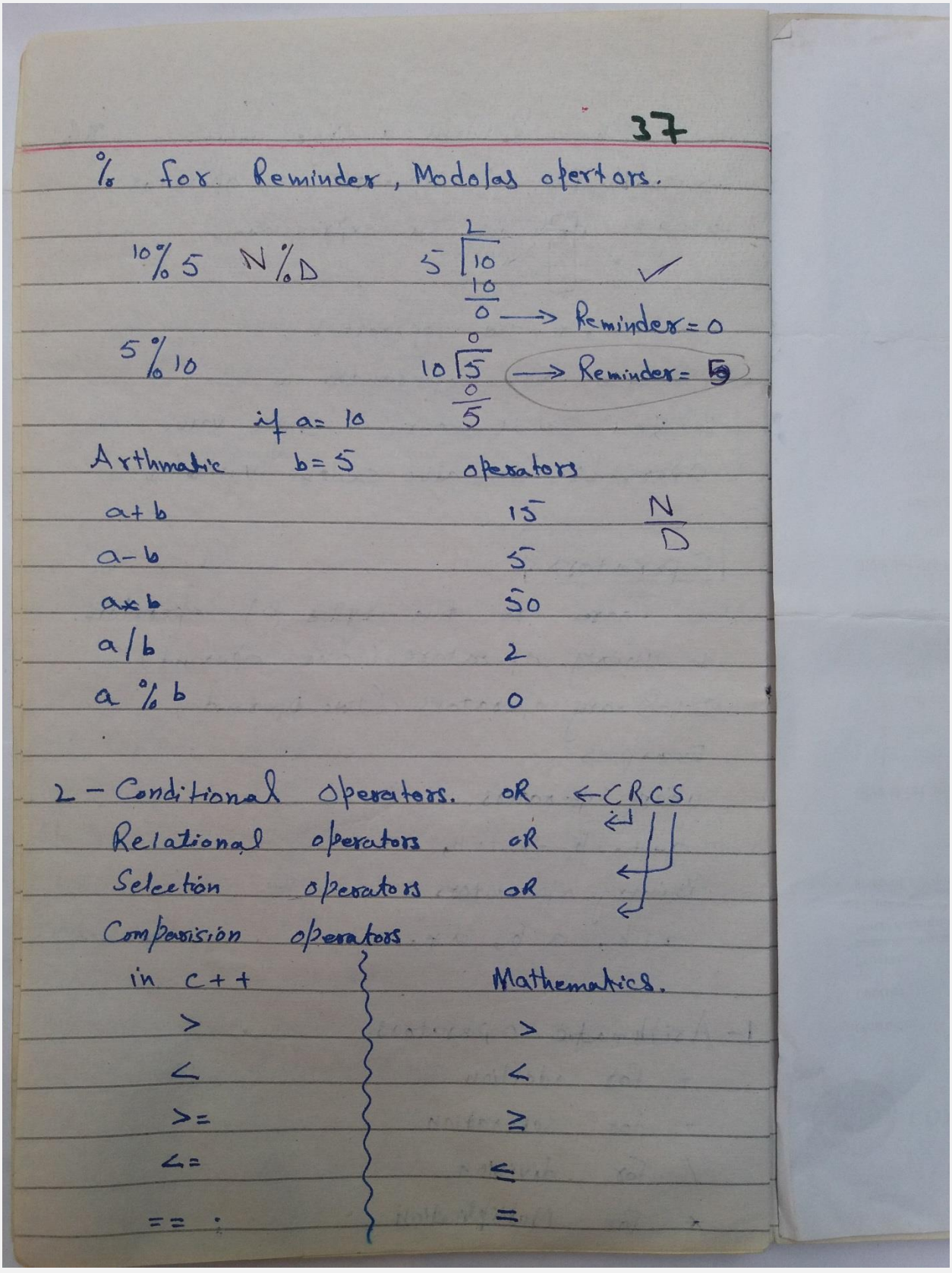
1- Arithmatic operators.

+ for addition

− for sobtration

/ for division

* for Multiplication



37

% for Reminder, Modolas opertors.

10 % 5 N % D

$$5\overline{)10}\ \ 2,\ 10,\ 0$$ ✓

0 → Reminder = 0

5 % 10

$$10\overline{)5}\ \ 0,\ 5,\ 0$$ → Reminder = 5

if a = 10, b = 5

Arthmatic	operators	N / D
a + b	15	
a - b	5	
a x b	50	
a / b	2	
a % b	0	

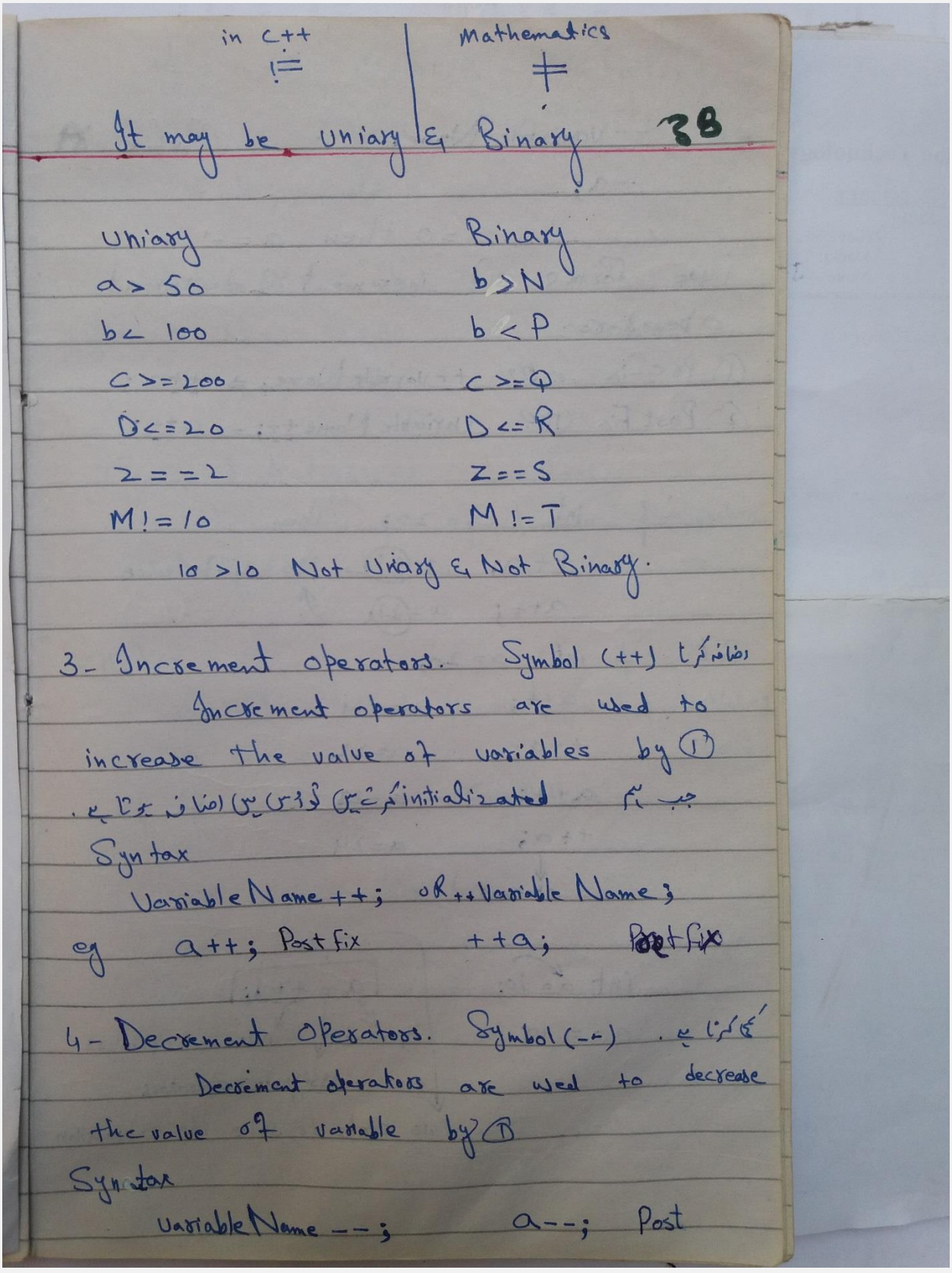
2 - Conditional operators. OR ← CRCS

Relational operators OR

Selection operators OR

Comparision operators

in C++	Mathematics.
>	>
<	<
>=	≥
<=	≤
==	=



in C++ != | Mathematics ≠

It may be Uniary & Binary 38

Uniary	Binary
a > 50	b > N
b < 100	b < P
C >= 200	C >= Q
D <= 20	D <= R
Z == 2	Z == S
M != 10	M != T

10 > 10 Not Uniary & Not Binary.

3- Increment operators. Symbol (++) ہوتا ہے

Increment operators are used to increase the value of variables by (1) جب کہ initialized کیا ہوا ہو۔

Syntax

VariableName++; OR ++VariableName;

eg a++; Post Fix ++a; Post Fix

4- Decrement operators. Symbol (--) ہوتا ہے۔

Decrement operators are used to decrease the value of variable by (1)

Syntax

VariableName--; a--; Post

--Variable Name; Prefix 39

--a;

a = 0 then a = -1

Two forms of increment & decrement operators.

1. Prefix ++Variable Name; ⇒ ++a;
2. Post Fix Variable Name++; ⇒ a++;

if int a = 20; Then

++a; a = 21 → No difference

a++; a = 21

But if int a = 20;

a++; a = 21

++a; a = 22

a++; a = 23

++a; a = 24

operand.

int a = 10;

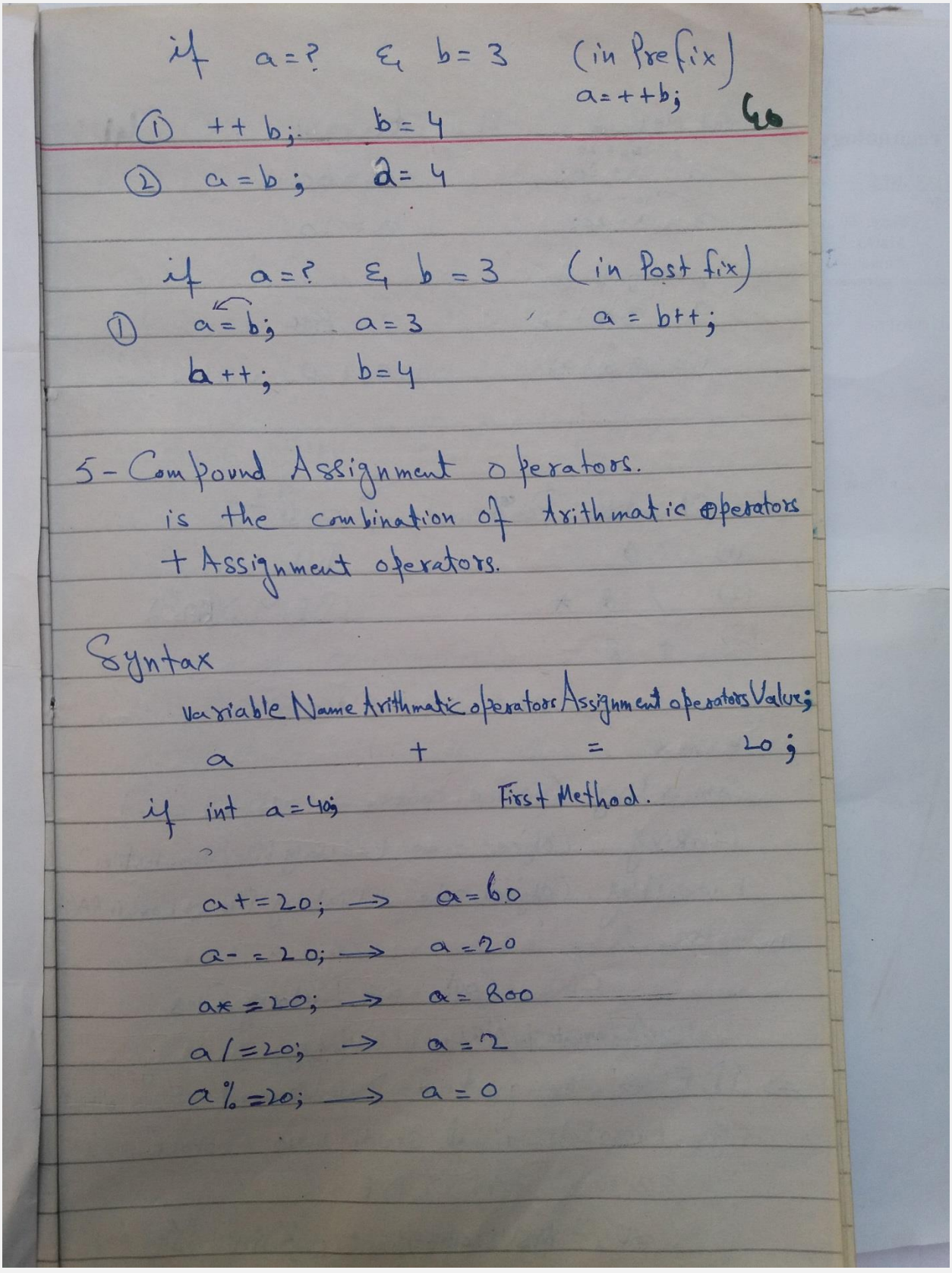
a = = 10;

a = ++b;

operand ↓ increment operators

Assignment operators

Two steps to solve the expression



if a=? & b=3 (in Prefix)

a=++b;

① ++b; b=4

② a=b; a=4

if a=? & b=3 (in Post fix)

① a=b; a=3 a=b++;

b++; b=4

5- Compound Assignment operators.

is the combination of Arithmatic operators + Assignment operators.

Syntax

Variable Name	Arithmatic operators	Assignment operators	Value;
a	+	=	20;

if int a=40;

First Method.

a+=20; → a=60

a-=20; → a=20

a*=20; → a=800

a/=20; → a=2

a%=20; → a=0

2nd Method. if int a=40; 41

$a = a + 20;$ (40 + 20) $a = 60$

$a = a - 20;$ (40 − 20) $a = 20$

$a = a * 20;$ (40 * 20) $a = 800$

$a = a \% 20;$ $a = 0$

6- Operators Precedence. → پیشی روی

(1) ( ) Rule

(2) / & * (P),(E),(M D),(A S)

(3) + & -

جب ہم پروگرام لکھتے ہیں تو اس کی تین Terms ہے.

Terms.

Compiling (Source code → object code)

Linking (object code + Liabary file / Headerfiles)

Executing (object code + Liabary file → Load in RAM)

Cint , Cout are Liabary files

رابطہ قائم کرنا / Communicate / ترقی اپنانا ماحول

⇒ IDE → Integrated Development Environment

eg MicroSoft visual studio 2010 Express Eidition is IDE

سوفٹ ویئر کو IDE کہتے ہیں جس طرح MSVS2010EE نظر آتا ہے.

=> Out Put Using [cout] -> Keyword 42

Robert Lafore is best book ↵

```
Cout << "Robert Lafore is best book";
```

↳ console output ↳ String constant

C++ میں output پر جو سکرین یا window نظر آتی ہے وہ (Console window) یا سکرین کہتے ہیں ۔

(Console: پیش کرنا / جو دکھائی دیتا)

کام انتہا ... اندراج

<< Insertion operators (OR) "Put to" operators

String is the combination of Multipals charaters

Constant => quatity which cannot change.

eg

```
cout << "Date of birth # is 05-09-1990";
cout << 2000;
cout << "a 2000";
```

Program output Every Lanuage has its own age

```
#include <iostream>
Using namespace std;
Voild main( )
{
```

(ہیڈر فائل) #include -> Head file, Liabary file, Built in file.

#include <iostream> -> Preprocessor Directive.

Keywords.

Preprocessor directive is an instruction given to compiler before the exectution of actual Program.

Body of function -> {

43

```
cout << "Every Lanuage has its own age";
Cin.get( );
        }
```

Body of function.

=> #include <math.h> & #include <graphic.h>

=> main (name of function) is Keyword. or build in Funtion.

سب سے Compiler ← main function کو جائے گا اور پہلی دو لائنز کو چھوڑ دے گا۔ بریکٹ کے ذریعے اگلے سٹیپ میں داخل ہو گا۔

پھر cout کو دیکھے گا اور ہم cout << "Every Lanuage has its own age"; کو show کرے گا اور cin.get(); ← output کو سکرین پہ show کو دکھے گی۔

Compiler ← white space کو ignore کر دے گا۔

iostream → flow of data.
→ output
→ input

White space is the blank space.

iostream اور using namespace std; یہ دونوں Compiler سے Request کرنے کے لیے استعمال ہوتے ہیں کہ ہمیں ہدایت کرنا

instructs the preprocessor to include a section of standard C++ code, know as a header iostream, that allows to perform standard input & output operations.

The extension of Header File is (.h)

Header files are the collection of standard liabrary function to perform different tasks.

There are many header file for dif purposes.

Comment => Two types of comments. 44

(1) Single line Comment. جب Comment ایک لائن کا ہو۔

(2) Muti-line comment. جب Comment ایک سے زیادہ لائن کا ہو۔

(i) Single line Comment.

use two Forward slash //

e.g. { A=3.14*r*r; // This is formula for Aeara= $A=\pi r^2$

Compiler always ignore the comment & didnot store it

(ii) Muti-line comment. use as /* */

e.g. { A=3.4*r*r; /* This is formula for Aear i.e. $A=\pi r^2$ of a circle*/

Program

```
#include<iostream>
using namespace std;
void main()
{int a,b,c;   2nd way int a=10, b=5, c;
cout<<"Enter the 1st value";
cin>>a;
cout<<"Enter the 2nd value";
cin>>b;
c=a*b;
cout<<"Result of multiplication"<<c;
cinget();}
```

Put to operater insertion

Extraction/ Get from operaters

45

2nd way.

```
cout << "Result of Multiplication";
cout << c;
cin.get();
}
```

=> Errors in Programs / Types of Errors in Program.

Three types of error maybe in a Program.
① Syntax ② Logical ③ run time error / Bug

Bug یعنی (An error in a Program)

run time error وہ ہوتا ہے جس میں data type کے مطابق data نہ ہو۔

=> Logical error is the most difficult to find.

=> Finding & removing errors is called Debugging.

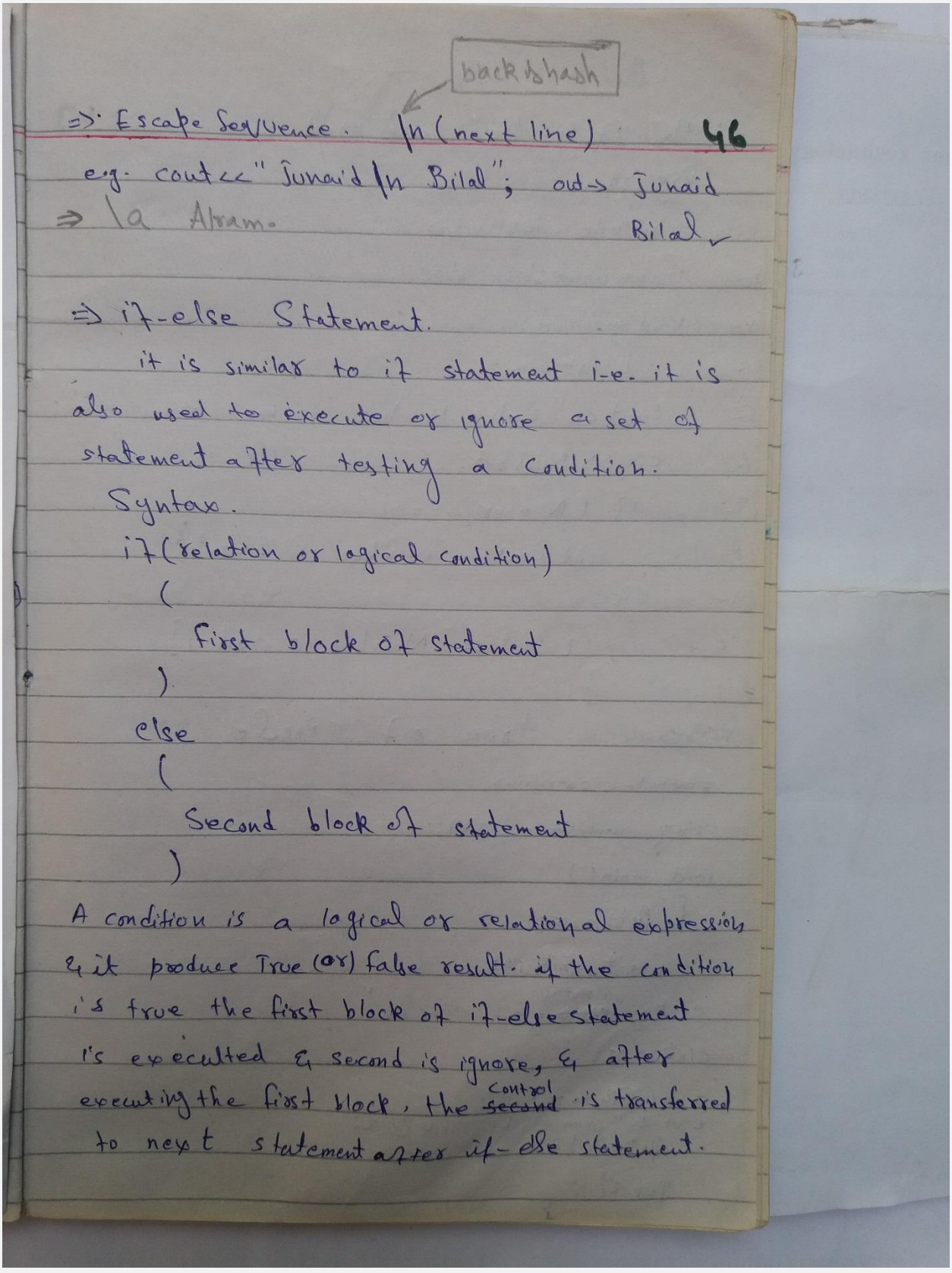
=> Manipulators (خوش اسلوبی سے استعمال کرنا) in C++

endl -> is a keyword. & used for ending the line. & meaning is End of Line.

an othe manipulator is setw

```
cout << "Junaid Bilal";        out -> Junaid Bilal
cout << "Junaid" << endl << "Bilal";   out -> Junaid
                                          Bilal
```

cout کے ساتھ endl کی بجائے \n بھی استعمال کر سکتے ہیں۔



backslash

=> Escape Sequence. \n (next line) 46

e.g. cout<<" Junaid \n Bilal"; out-> Junaid
Bilal

=> \a Alram.

=> if-else Statement.

it is similar to if statement i-e. it is also used to execute or ignore a set of statement after testing a condition.

Syntax.

```
if(relation or logical condition)
  (
    First block of statement
  )
else
  (
    Second block of statement
  )
```

A condition is a logical or relational expression & it produce True (or) false result. if the condition is true the first block of if-else statement is executed & second is ignore, & after executing the first block, the ~~second~~ control is transferred to next statement after if-else statement.

both if & else are used for result. 47

Program.

```
1- #include <iostream>
2- using namespace std;
3- Void main()
4-     { int num;
5-  cout<<" Enter Number to check Even or odd";
6-  cin >> Number;
7-     if (Number%2==0)
8-         {cout << "Number is Even";}
9-  else
10-        {cout<<" Number is odd";}
11-    }
```

Program Area of circle.

```
#include<iostream>
using namespace std;
Void main()
  { float a, r;
    float Pi=3.14;
cout<< "Enter the radius of circle"
cint>> r;
a =(Pi * r * r);
cout << " area of the circle"<<a;
   getch();
   getch();  }
```

48

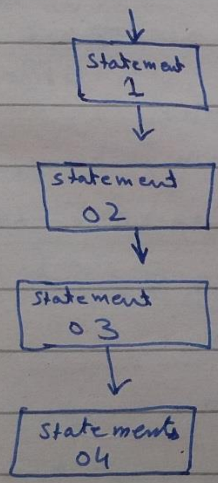
=> Control Statements. The flow of control jumps from one part of the program to another, depending on calculation performed in the program. The program / statements that causes such jumps are called control statements. [illegible] statements 09 [illegible]

There are 4 types of Control Statements.

① Sequence Statement
② Selection Statements
③ Iteration Statements.
④ Funtion Call.

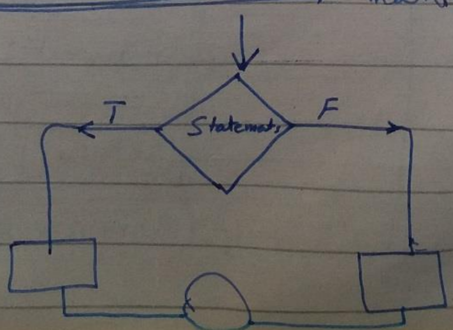
① Sequence Statement:-

In Sequence Statements instructions / Statements are executed on the basic of Sequence / oder.



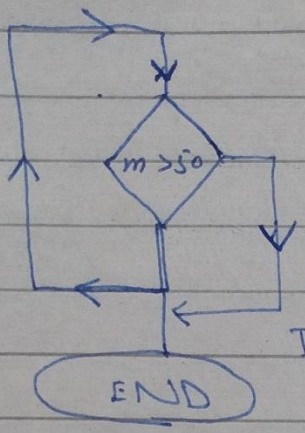
② Selection Statements.

In Selection Statements Instruction / Statements are executed on the basic of dision / selection OR condition.
It is also know dision making statement.



49

(3) Interation / Loop / Repeated.



out Put
C++
C++
C++
C++
C++

In Interation or Loop or Repeated Statement instration / statements are executed on the basic of repetition. Repetition is continues while a condition is true. Three types of loop (1) For loop (2) The while loop (3) The do loop

(4) Funtion - Call.

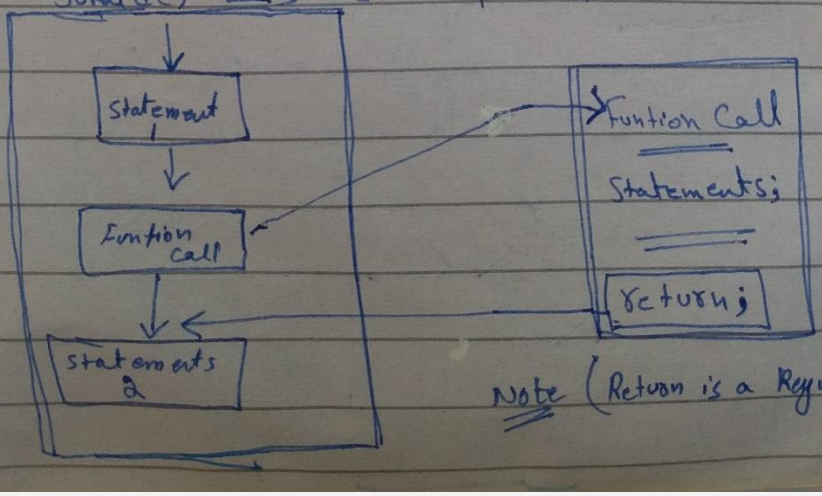
There are two types of funtion Call.

① Stantard funtion ② user define Funtion.

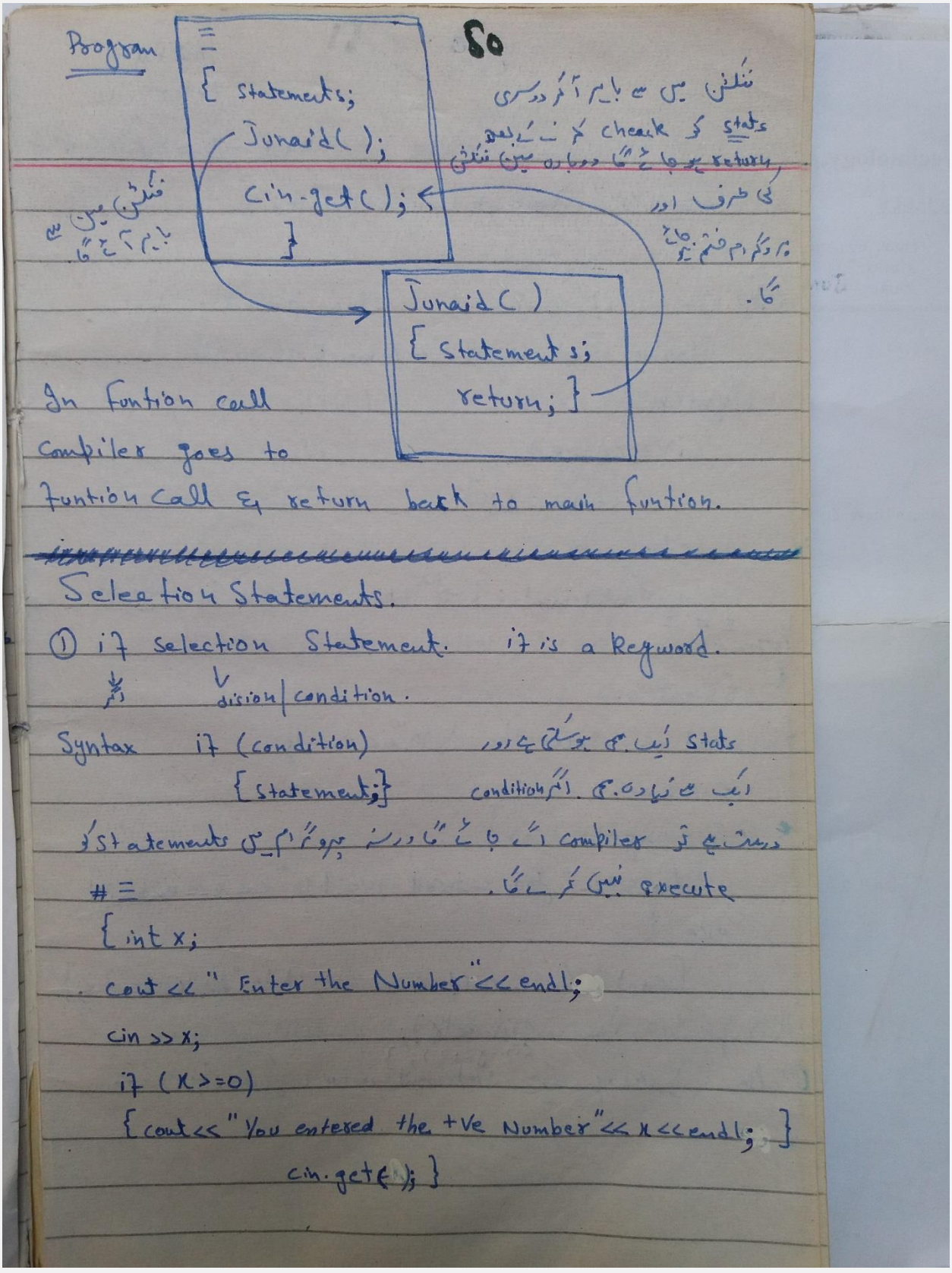
Bulit in ہے جو Compiler / C++ کے ساتھ آتے ہیں۔

Junaid ⟶ is a variable.

Junaid() ⟶ is a funtion.



Note (Return is a Reyword)



80

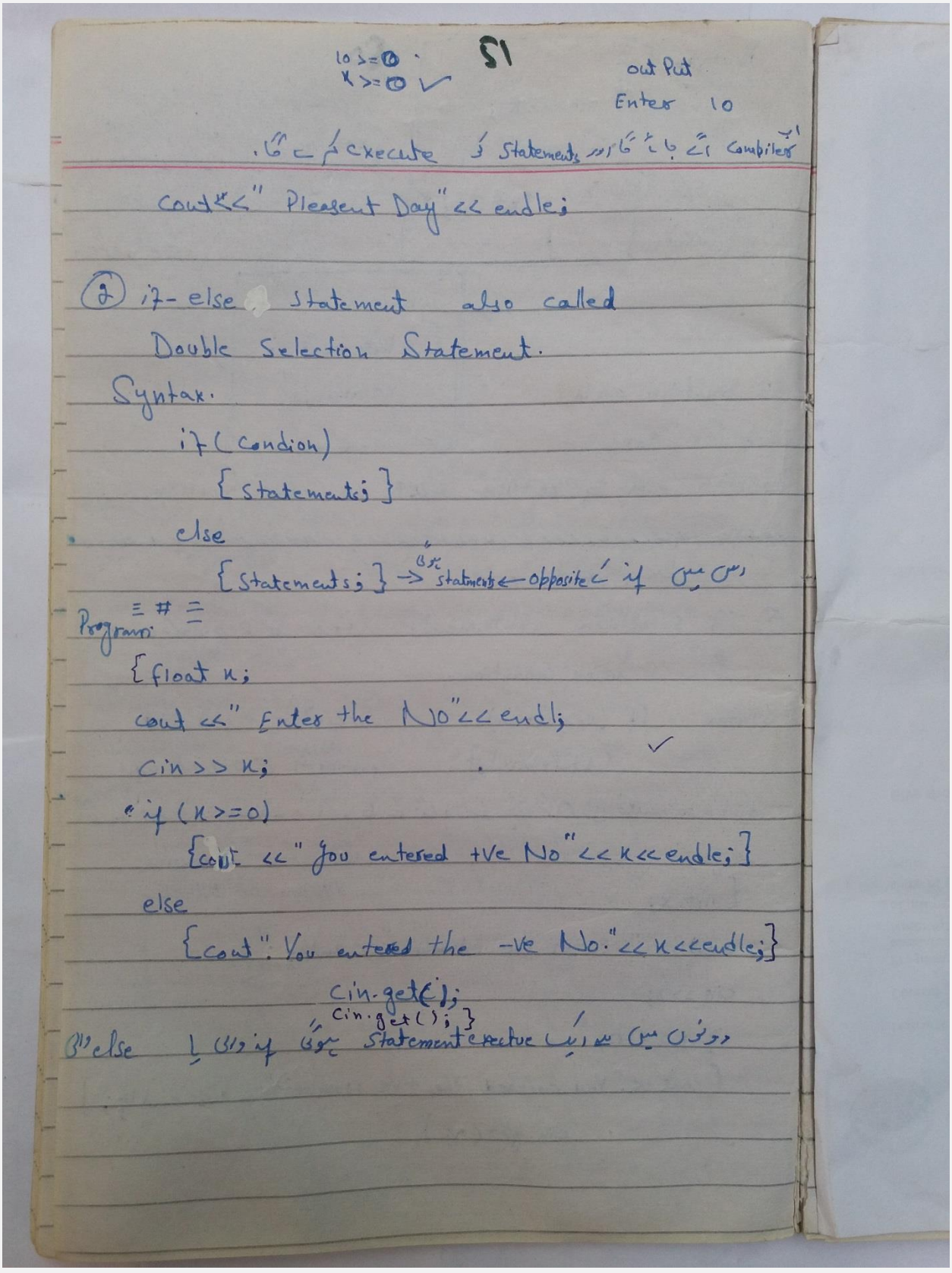
In funtion call compiler goes to funtion call & return back to main funtion.

Selection Statements.

① if selection Statement. if is a Keyword.

Syntax if (condition)
{Statement;}

#=
{int x;
cout << "Enter the Number" << endl;
cin >> x;
if (x>=0)
{cout << "You entered the +ve Number" << x << endl; }
cin.get( ); }



51

10 >= 0 ✓
x >= 0 ✓

out Put
Enter 10

Compiler ہر Statements کو execute کرتا ہے۔

cout<<" Pleasent Day"<<endl;

② if- else Statement also called
Double Selection Statement.

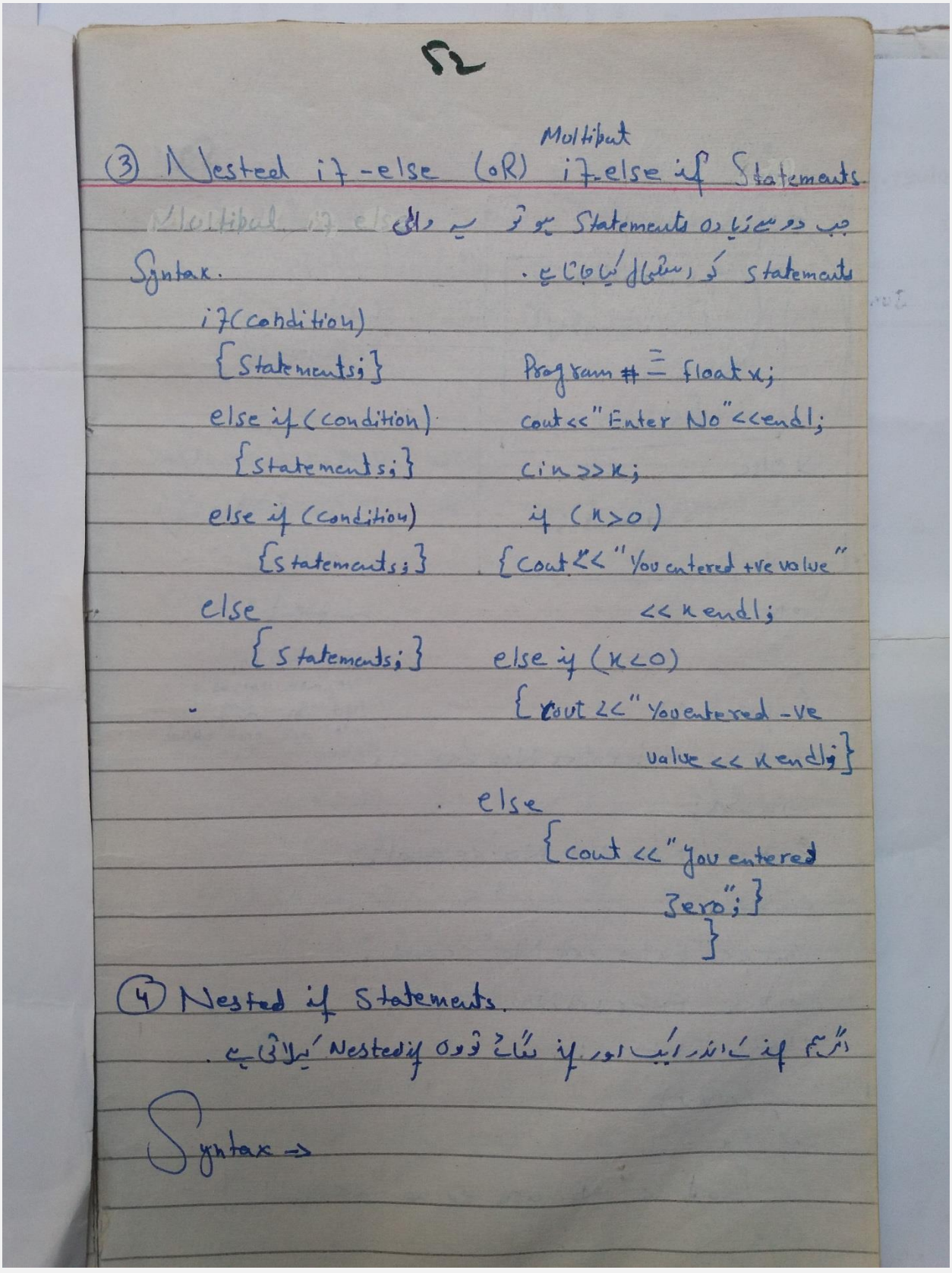
Syntax.

if (condion)
{statements;}
else
{statements;} → statements ← opposite کی if میں

Program:

{float x;
cout <<" Enter the No"<<endl;
cin >> x;
if (x>=0)
{cout <<" You entered +Ve No"<<x<<endl;}
else
{cout<<" You entered the -Ve No."<<x<<endl;}
cin.get();
cin.get();}

If else میں if کی statement execute ہوتی ہے دونوں میں سے ایک



52

③ Nested if-else (OR) if-else if Statements (Multibut)

جب دو سے زیادہ Statements ہو تو یہ والی Syntax کو استعمال کیا جاتا ہے۔

```
if(condition)
 {Statements;}
else if (condition)
  {statements;}
else if (condition)
  {statements;}
else
  {statements;}
```

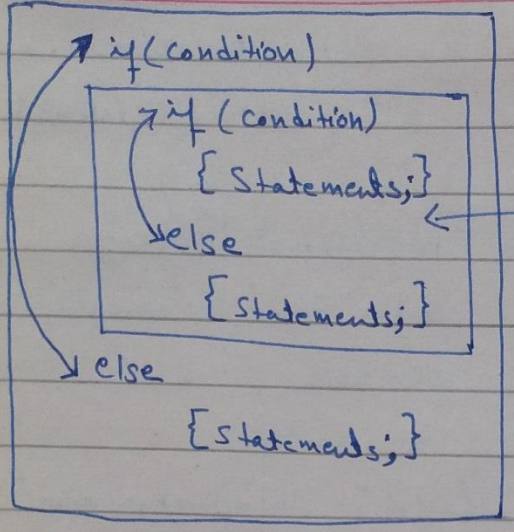
```
Program # = float x;
cout<<"Enter No"<<endl;
cin>>x;
if (x>0)
{cout<<"You entered +ve value"
<<x endl;
else if (x<0)
{cout<<"You entered -ve
value<<x endl;}
else
{cout<<"you entered
zero";}
}
```

④ Nested if Statements.

اگر if کے اندر ایک اور if بنائے تو وہ Nested if کہلاتی ہے۔

Syntax ->

53



Program:

```
{ if a, b, c;
cout << " Enter 1st No. " << endl;
cin >> a;
cout << " Enter 2nd No. " << endl;
cin >> b;
cout << " Enter 3rd No. " << endl;
cout << " Enter 3rd No. " << endl;
cin >> a >> b >> c;
if (a == b)
if (a == c)
   {cout << " No. are equal " << endl; }
else
   {cout << " No. are Different " << endl; }
```

(OR)

output

```
Enter 3 No.
No are equal
No are dif
No are Not equal
```

else 54

```
{cout << " No are Different " << endl;}
  cin.get(); OR getchar();
  cin.get(); OR getchar();
  }
```

Increment operatorse.

++ increment اور decrement کو ترتیب دیتے ہیں جو variable (a) کے
e.g a++; OR ++a; ساتھ لگتے ہیں.

Program

```
Voidmain()
{ int count = 10;
cout << count << endl;
    count++;
    ++count;
cout << count << endl;
      cin.getch();
      cin.getch();
      }
```

Output

10
12

Note پروگرام میں جتنے cout ہونگے اتنے ہی Result شو ہونگے. اوپر والے میں ② cout ہے اس لیے output رزلٹ میں بھی دو cout آئے ہیں

Program. 55

```
{int count=20;                          output
cout << count << endl;                  20
cout << ++count << endl;                21
cout << count << endl;                  21
cout << count++ << endl;                21
cout << count << endl;                  22
                getchar();
                getchar(); }
```

Logical operator?

There are three logical operators in C++

① AND — &&
② OR — ||
③ NOT — !

→ No space b/w them.

Syntax. AND & OR

((Condition 1) && (condition 2)) (OR ||)

(Condition 1 — space — && — space — condition 2)

Note → Compiler Read کرتا ہے space جو condition کے بیچ میں ہو

Syntax NOT operater ! 86

(!(condition))

Note

NOT operater !

not equal to !=

AND &&

OR ||

Table.

condition No. 1	condition No. 2	AND &&	OR ||
True	True	output	output
True	False	X	output
False	True	X	output
False	False	X	X

Program. =

```
void main()
{int x,y,z;
cout << "Enter 3 No" <<endl;
cin >> x y>>z;
if((x>y) &&(x>z))
{cout<< "Maximun No." <<x<<endl;}
else if
     ((y>x) &&(y>z))
{cout << "Maximun No" <<y <<endl;}
else
  {cout << "Max No" <<z endl;}
```

out Put.

10, 20, 30

max No is 30

```
cin.get();
    }
```

57

Program. ≡ Not Program.

```
void main()
{int m;
cout<<"enter marks"<<endl;
cin>>m;
 if (!(m>50))
  { cout << "False"; }
 else
    {cout << "Pas"; }
          cin.get(); }
```

Assignment Program.

output is

Press 1 for Addition.
 " 2 " Subtration.
 " 3 " Multiplication
 " 4 for Division.

Press Enter 1st value
 Enter 2nd value.

Press (5) show invalid No.

Hind choice → variable.

swith, Nested if, Nested if else.

choice میں value کو show کروانا ہے

Program. 58

```
#include<iostream>
Using namespace std;
Void main( )
{ int a, b, choice, sum, sub, multi, div;
cout << " Enter the 1st number " << endl;
cin >> a;
cout << " Enter the 2nd number " << endl;
cin >> b;
cout << " Enter your choice " << endl;
cin >> choice;
    { if (choice == 1)
        { sum = (a+b);
        cout << "sum is " << sum << endl; }
      else if ( choice == 2)
        { sub = (a-b);
        cout << " sub is " << sub << endl; }
      else if (choice == 3)
        { div = (a/b);
        cout << " div is " << div << endl; }
      else if (choice == 4)
        { Multi=(a*b);
        cout << multi is " << multi << endl; }
      else
    { cout << " You enter the invalid No " << endl; } }
```

getchar();
getchar();

Cout Put

Assignment

59

}

Program.

Sessional-I Question

```
#include<iostream>
Using namespace std;
Void main()
{ int count = 99;
  cout << count << endl;
Cout << ++count << endl << count++ << endl;
cout << count << endl;
Cout << -- count << endl << count-- << endl;
cout << count << endl;
      cin.get();
      cin.get();
    }
```

99+1 = 100

101

Dry run

99
99+1 = 100
++100 = 101
101-1 = 100
1-100 = 99

output 99, 101, 99, 101, 99, 101, 99

⇒ Operater Precedance.

++ Prefix Postfix ++ Precedance

Compiler C++

Program given as example above.

Program. Sessional -1

≡ Question 6

```
int p=1, q=9;
if((p>=3) || (q<11 && p+q<15))
  {cout << "Best of Luck"<<endl;}
else
  {cout << " Good Luck" <<endl;}
   getchar();   | cin.get();
   get char();  | cin.get();
}
```

(E) (T) (T) (T) (OR) F || T = out put ✓

Program Coding Increment & decrement.

≡ According to Precedance.

```
{ int bilal=7;
  cout << bilal<<endl;
  cout<< bilal++<<endl<< ++bilal <<endl;
  cout << bilal <<endl;
  cout <<- bilal <<endl<< bilal --<<endl;
  cout << bilal <<endl;
  cout << ++bilal <<endl<< bilal--<<endl;
  cout << bilal <<endl;
  cout<< -bilal <<endl<< bilal++ << endl;
  cout << bilal <<endl;
   getchar();
   get char(); }
```

out Put

7, 8, 9, 9, 7, 7, 7, 7, 7, 7, 7

7

61

Switch Multiple Selectio Statement

Program.

```
#include<iostream>
Using name space std;
Void main()
{
  char bilal;
  float a,b,c;
  cout<<"+ and j for addition"<<endl<<"- for subtration"<<
      endl<<"* for multiplication"<<endl<<"/ and k
      for division"<<endl;
  cout<<"enter the operater +,-,*,/,j,k"<<endl<<endl;
  cin>>bilal;
  cout<<"enter the operands "<<endl;
  cin>>a;
  cout<<"enter the 2nd operands "<<endl;
  cin>>b;
  Switch(bilal)
  { Case '+':
    case 'j':
    c=a+b;
    cout<<"result of addition"<<c<<endl;
    break;
```

Switch اور Case دونوں کا

Data Type ← Same ہونا چاہئے

Page 1073

61A FA14-BSSE-007

Switch Multiple Selection Statement (OR)

* Switch Statement in C++ Programming.

* Switch statement compares the result of a single expression with multiple cases. Switch is also a control structure & it is used to select one option from a set of options.

* Consider a situation in which, only one block of code needs to be executed among many blocks. This type of situation can be handled using nested if-else statement but, the better way of handling this type of problems is using switch... case statement.

* It compares the value of an expression or a variable against a list of cases.

* The case labels & value of expression or variable must be an integer or a character.

* It must not be a float or double value, if the value of expression in switch is float or double type then the compiler will generate error message.

* Syntax of Switch Statement.

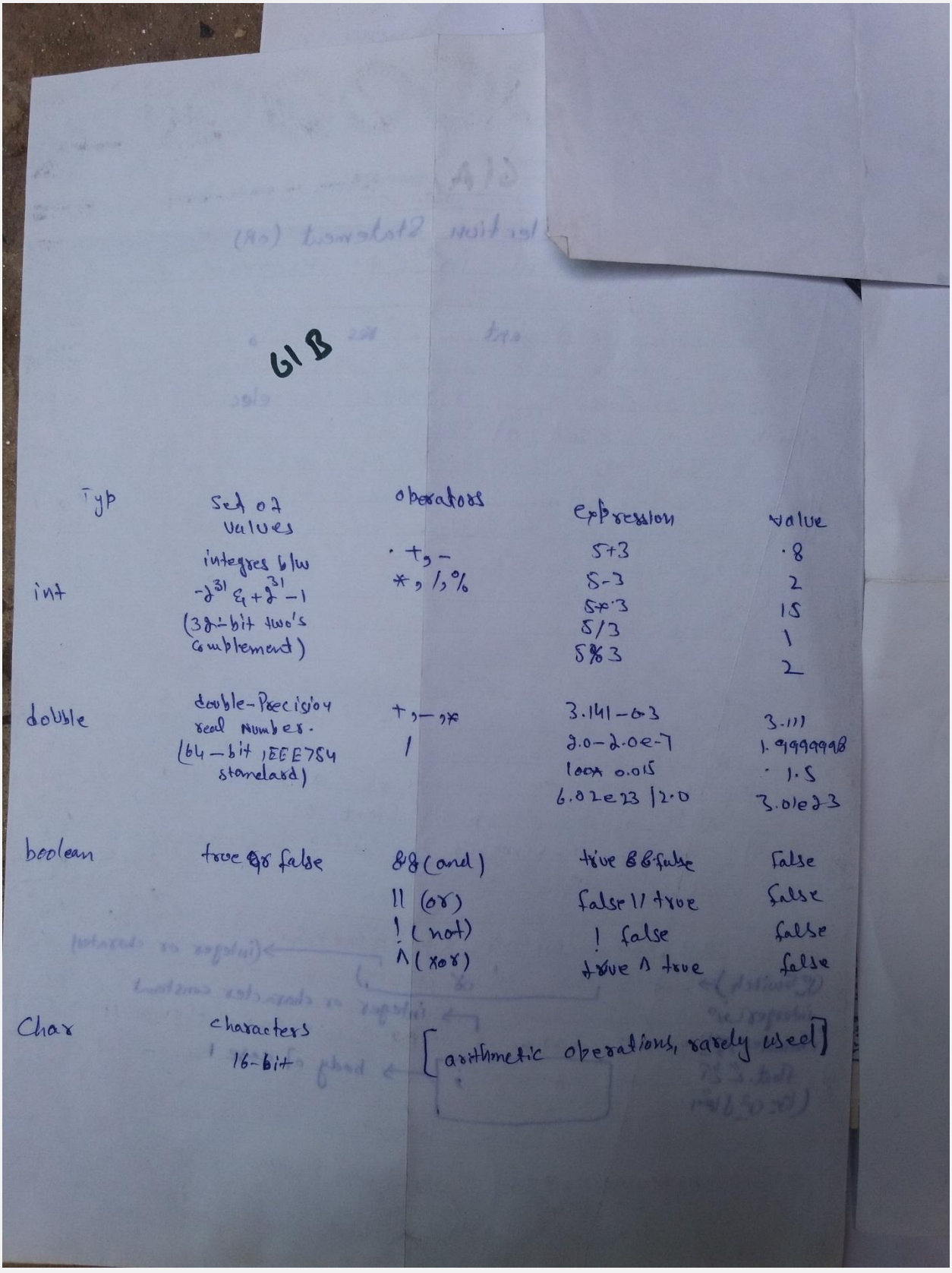
(integer or character)

switch(expression or variable)
{
case val-1 : // use colon not semi colon
Statement-1;
break;
case val-2 : // use colon not semi colon

integer ✓
character ✓
float ✗

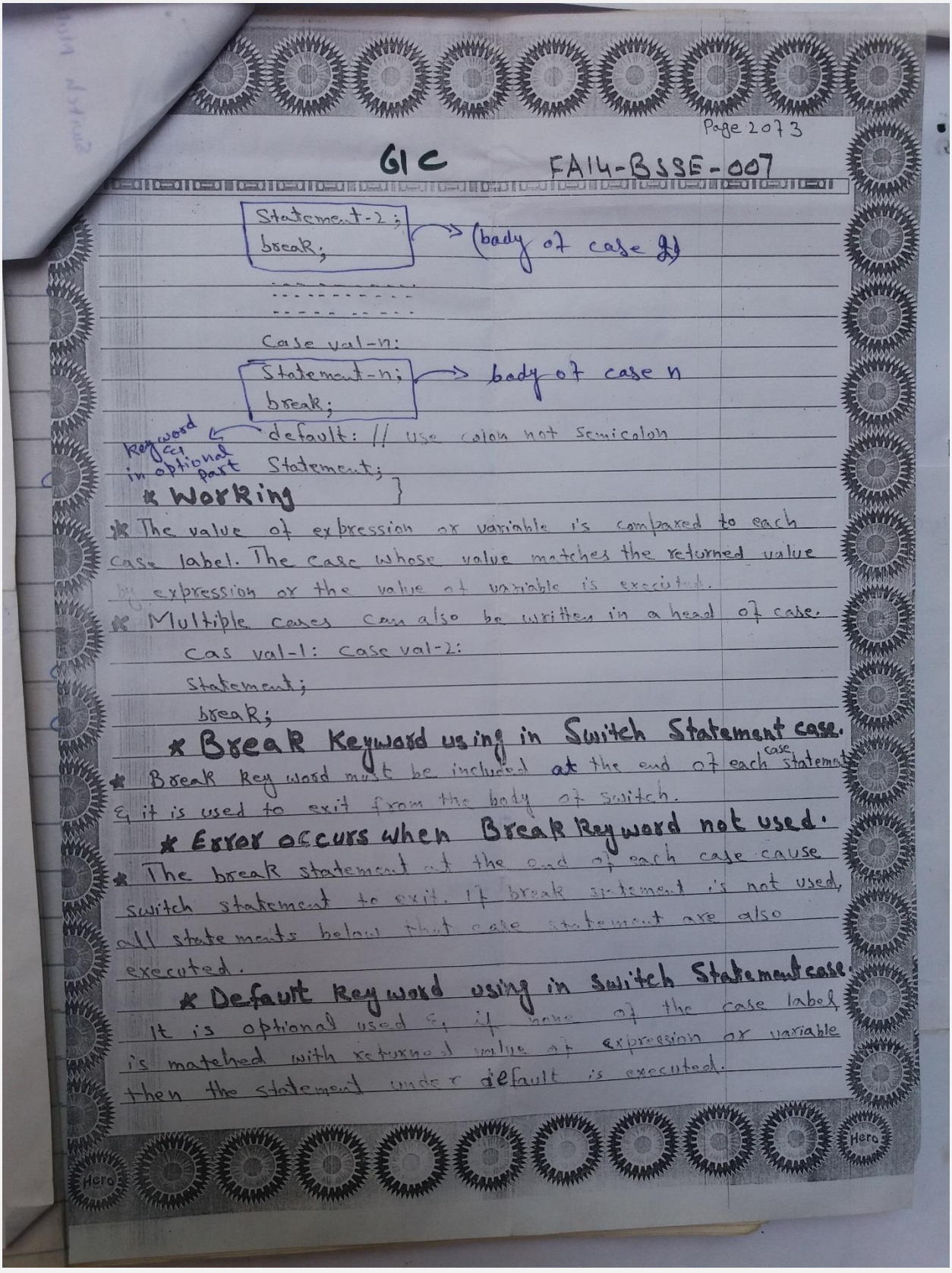
integer or character constant

body of case 1



61B

Typ	Set of Values	operators	Expression	value
int	integres b/w -2^{31} & $+2^{31}-1$ (32-bit two's complement)	+, -	5+3	8
		*, /, %	5-3	2
			5*3	15
			5/3	1
			5%3	2
double	double-Precision real Numbers. (64-bit IEEE754 standard)	+, -, *	3.141-03	3.111
		/	2.0-2.0e-7	1.9999998
			100* 0.015	1.5
			6.02e23 /2.0	3.01e23
boolean	true & false	&& (and)	true && false	false
		|| (or)	false || true	false
		! (not)	! false	false
		^ (xor)	true ^ true	false
Char	characters 16-bit	[arithmetic operations, rarely used]		



Page 2073

GIC FA14-BSSE-007

Statement-2;
break;
→ body of case 2

.....

Case val-n:
Statement-n;
break;
→ body of case n

Keyword is optional part
default: // use colon not semicolon
Statement;

*** Working**

* The value of expression or variable is compared to each case label. The case whose value matches the returned value by expression or the value of variable is executed.

* Multible cases can also be written in a head of case.

Cas val-1: case val-2:
Statement;
break;

*** Break Keyword using in Switch Statement case.**

* Break Key word must be included at the end of each case statement. E.g it is used to exit from the body of switch.

*** Error occurs when Break Keyword not used.**

* The break statement at the end of each case cause switch statement to exit. If break statement is not used, all statements below that case statement are also executed.

*** Default Keyword using in Switch Statement case.**

It is optional used E.g if none of the case label is matched with returned value of expression or variable then the statement under default is executed.

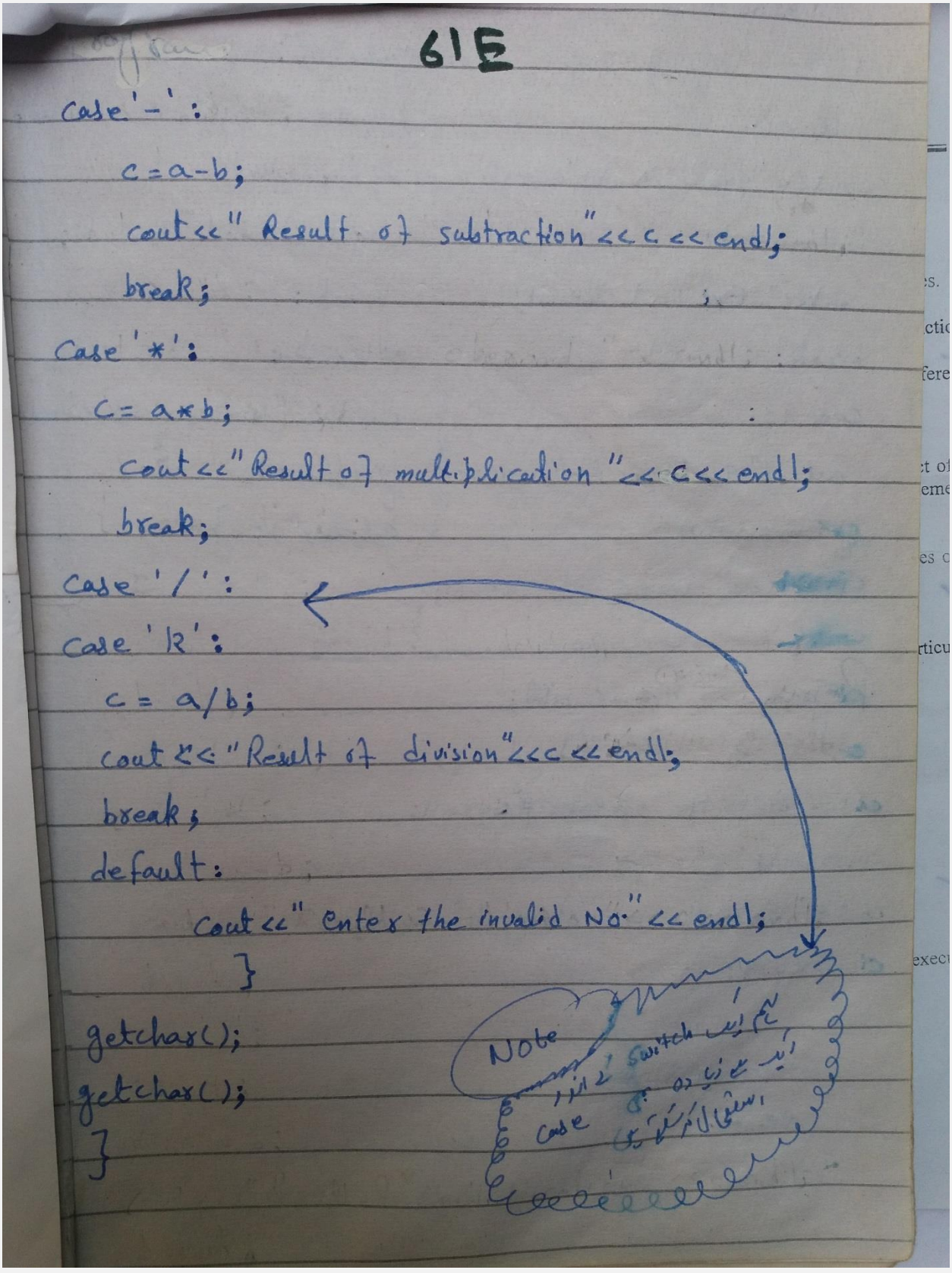
61 D FA14-BSSE-007

★ The default Keyword is not fixed. It may be placed before the first case or after the last case.

★ Switch Statement Program in C++

```
#include<iostream>
Using namespace std;
Void main()
{ char ch;
 cout<<"Enter a charactor to cheak it is Vowel or not"
 cin>>ch;
    Switch(ch)
        {
        Case 'a': Case 'A':
        cout<<ch<<" is a vowel";
        break;
        Case 'e': Case 'E':
        cout<<ch<<" is a vowel";
        break;
        Case 'i': Case 'I':
        cout<<ch<<" is a vowel";
        break;
        Case 'o': Case 'O':
        Cout<<ch<<" is a vowel";
        break;
        Case 'u': Case 'U':
        cout<<ch<<" is a vowel";
        break;
  default:
  cout<<ch<<" is a consonant character not a vowel";}
}
```

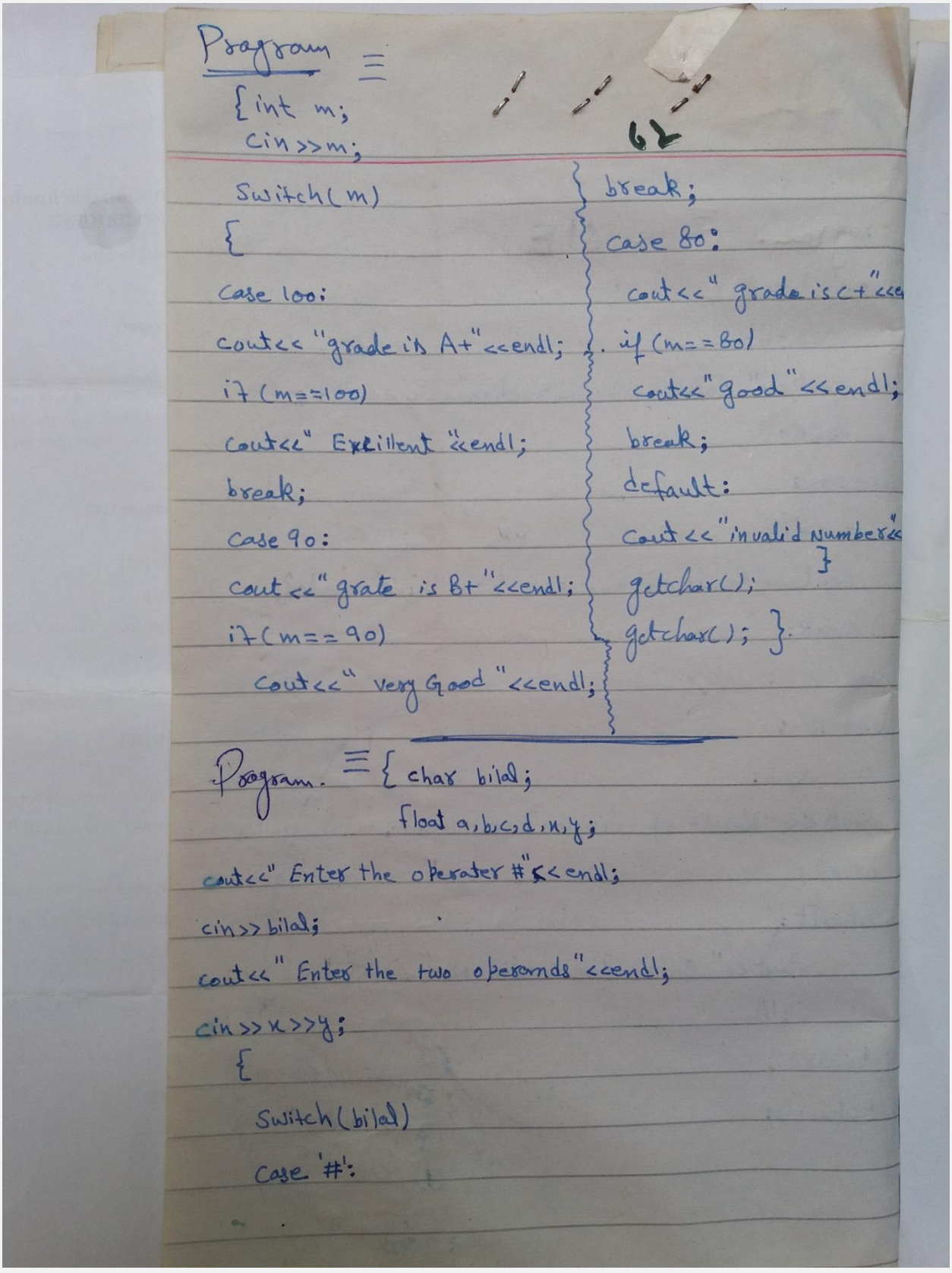
جو variable char یا int کا ہوگا وہی switch کا ہوگا ورنہ error آئے گا۔



61E

```
case '-':
    c=a-b;
    cout<<" Result of subtraction"<<c<<endl;
    break;
case '*':
    c= a*b;
    cout<<" Result of multiplication "<<c<<endl;
    break;
case '/':
case 'k':
    c = a/b;
    cout<<" Result of division"<<c<<endl;
    break;
default:
    cout<<" enter the invalid No."<<endl;
    }
getchar();
getchar();
}
```

Note

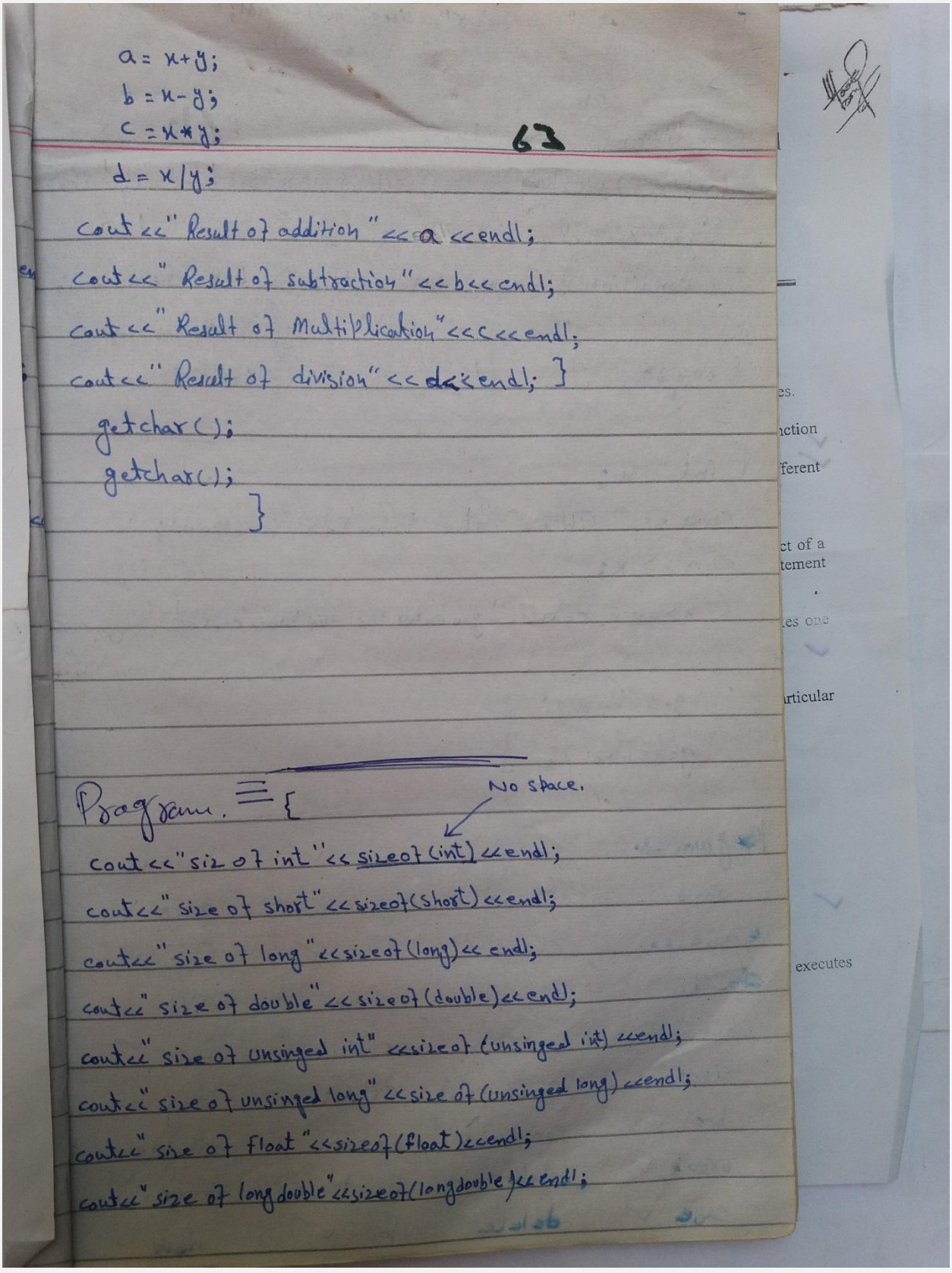


Program =

```
{int m;
 cin>>m;
 Switch(m)
 {
case 100:
cout<< "grade is A+"<<endl;
if (m==100)
Cout<<" Exellent "<<endl;
break;
case 90:
cout <<" grate is B+"<<endl;
if(m==90)
  Cout<<" very Good "<<endl;
break;
case 80:
 cout<<" grade is C+"<<
. if (m==80)
 cout<<"good"<<endl;
 break;
 default:
 Cout <<"invalid number"<<
 }
 getchar();
 getchar(); }.
```

Program. =

```
{ char bilal;
  float a,b,c,d,x,y;
cout<<" Enter the operater #"<<endl;
cin>>bilal;
cout<<" Enter the two operands"<<endl;
cin>>x>>y;
  {
   Switch(bilal)
   Case '#':
```



```
a = x+y;
b = x-y;
c = x*y;                                    63
d = x/y;
cout<<" Result of addition "<<a<<endl;
cout<<" Result of subtraction"<<b<<endl;
cout<<" Result of Multiplication"<<c<<endl;
cout<<" Result of division"<<d<<endl; }
getchar();
getchar();
}
```

Program. = {

No space.

```
cout<<"size of int"<<sizeof(int)<<endl;
cout<<" size of short"<<sizeof(short)<<endl;
cout<<" size of long "<<sizeof(long)<<endl;
cout<<" size of double"<<sizeof(double)<<endl;
cout<<" size of unsinged int"<<sizeof(unsinged int)<<endl;
cout<<" size of unsinged long"<<size of(unsinged long)<<endl;
cout<<" size of float "<<sizeof(float)<<endl;
cout<<" size of long double"<<sizeof(long double)<<endl;
```

```
cout<<"size of char"<<sizeof(char)<<endl;
cout<<"size of boolean"<<sizeof(bool)<<endl;
getchar();
getchar(); }
```

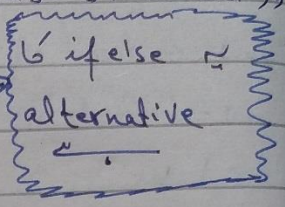
⇒

Conditional operaters 64

≈ (Same) working ✓ Condition operaters ,,, if-else

Syntax.

(condition ? True Statement : False Statement);



Program.

```
{ int x;
cout << " Enter the Number" <<endl;
cin >> x;
( x>=0 ? cout<< " you enter the +ve No": cout<< " you enter the -ve No.";
getchar();
getchar(); }
```

Keywords.			
and	catch	do ✓	friend
and_eq	char ✓	dynamic_cast	goto ✓
auto	class ✓	else ✓	if ✓
bitand	compl	enum ✓	inline
bitor	const ✓	explicit	int ✓
bool ✓	const_cast	export	long ✓
break ✓	continue ✓	extern	mutable ✓
case ✓	default ✓	float ✓	namespace
	delete ✓	for ✓	new ✓

not	reinterpret_cast	switch	typename	65
operator	return	temlate	union	
or	short	this	unsigned	while
or_eq	signed	throw	using	xor
Private	size of	true	virtual	xor_eq
Protected	Static	try	Void	
Public	Static_cast	typedef	Volatile	
register	struct	typeid	wchar_t	

Iteration (or) loop. (OR) repeatation.
Repeatation of loop is called Iteration.
There are three types of loop.
(1) for loop (2) while loop. (3) do while loop
Syntax. (دونوں سیمی کولن بیچ میں لگتے ہیں) (use only one operater at a time.)

① For (Initalization; Condition; Increment operater / decrement operater)
{ Statements; }

ایک ہی پروگرام میں ایک یا دو یا تین loops کا استعمال ہو سکتا ہے

Program ≡

```
{ int j;
  for(j=1; j<5; j++(OR) ++j)
  { cout<< j << "\t" << j*j<<endl; }
  getchar();
  getchar();
}
```

Dry run is basically is calculation behind C++

66

Dry Run — out Put

Dry Run		\t	out Put
j=1	True	\t	1
j=2	"	"	4
j=3	"	"	9
j=4	"	"	16
j=5	false	"	loop terminated

it is the program which show the squre after the tabe.

⇒ 2nd way. ہم loop کے اندر ہی initalized کرتے ہیں

```
for (int j=1 ; j<5 ; j++)
```

(space between int and j)

⇒ Program =

```
{ int j;
   for(int k=0 ; k<3 ; k++)
     {cout << k << "\t";
          j= k* k* k;
     cout << j << endl; }
     Getchar();
     getchar();
}
```

0 - 0
1 - 1
2 - 8

Q: write a Program that input a No. for table from the user & also input length for table. & show the table according to its length → Hint (use for loop)

No. 1

Program for any type of table

67

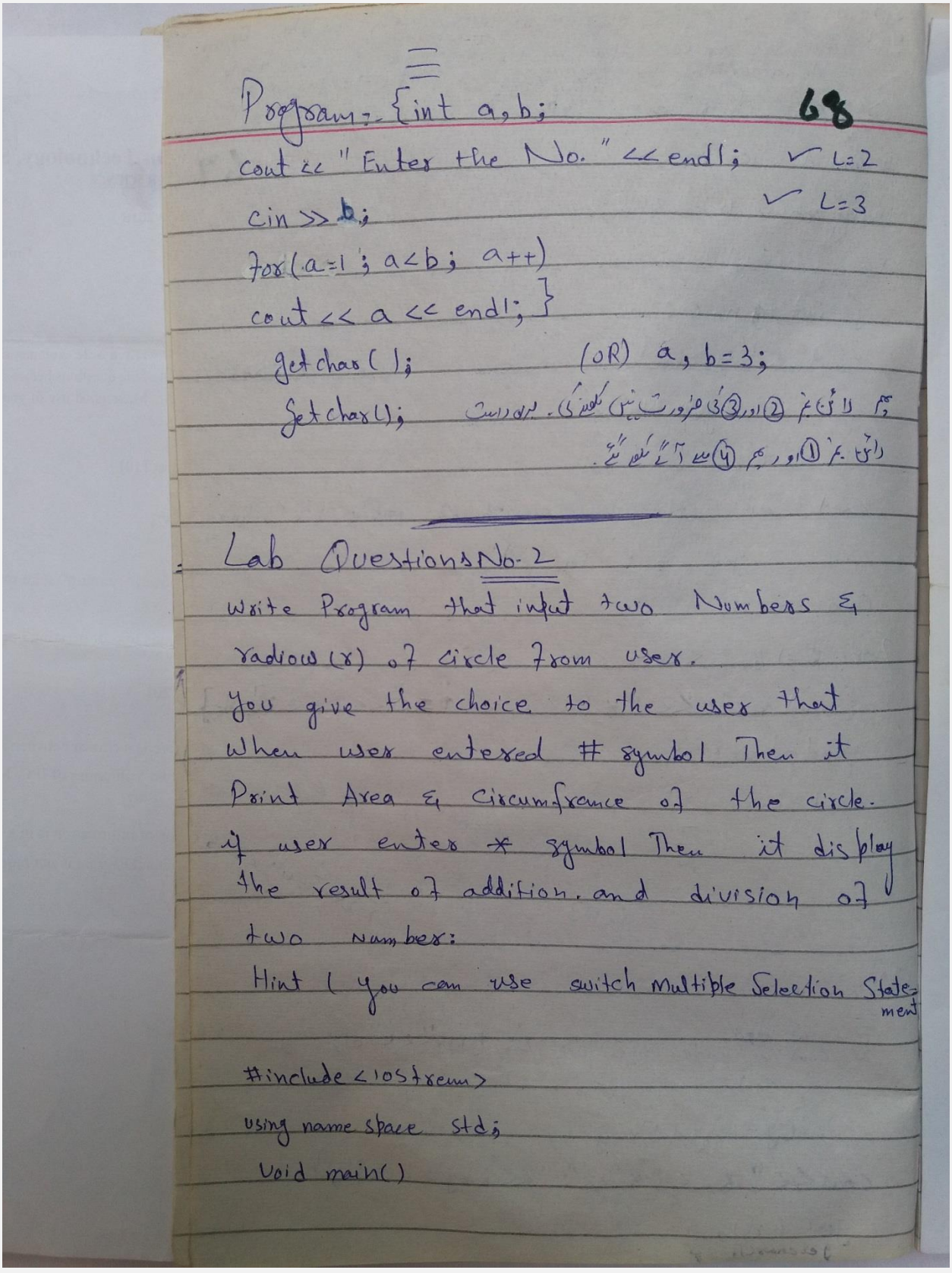
```
# include <iostream>
using namespace std;
Void main()
{ int t, n, c;
  Cout << " Enter the number of table \t (which
             you want)";
  cin >> t;
  Cout << " Enter the length of table \t (where
             you want to go)";
  Cin >> n;
 For( c=1; c<=n; c++)
  {cout << t << "*" << c << "=" << t*c << endl;}
  getchar();
  getchar();
            }
```

OW → endl استعمال [illegible] Table [illegible]

e.g 2*1=2 2*2=4 2*3=6

2nd way for Table.

```
{  int x, y;
   cout << "Enter the No. for table" << endl;
   cin >> x
   for(y=1; y<5; y++)
 cout << x << "*" << "=" << x*y << endl; }
   getchar();
   getchar();  }
```



Program:- {int a, b;

cout << "Enter the No." << endl;

cin >> b;

for(a=1; a<b; a++)

cout << a << endl; }

getchar();

Getchar();

68

✓ L=2

✓ L=3

(OR) a, b=3;

اگر آپ نے ② اور ③ کی ضرورت نہیں کی۔ تو درست

آپ نے ① اور ④ سے آگے نہیں بڑھے۔

Lab Questions No. 2

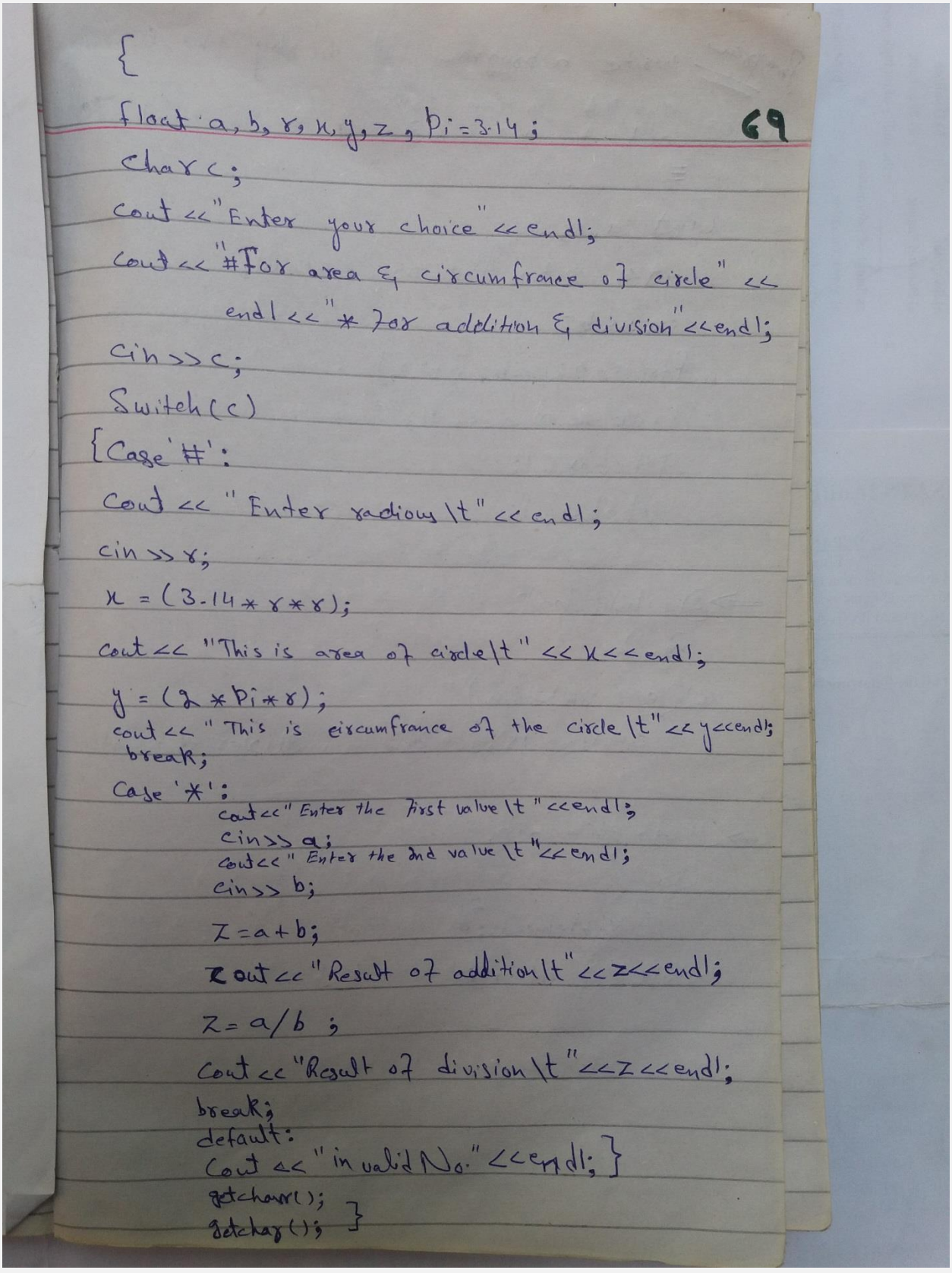
Write Program that input two Numbers & radious (r) of circle from user.

you give the choice to the user that when user entered # symbol Then it Print Area & Circumfrance of the circle.

if user enter * symbol Then it display the result of addition. and division of two Number:

Hint (you can use switch Multible Selection Statement

```
#include <iostream>
using name space std;
Void main()
```



```
{
float a, b, r, x, y, z, Pi = 3.14;                    69
char c;
cout << "Enter your choice" << endl;
cout << "#For area & circumfrance of circle" <<
        endl << "* for addition & division" << endl;
cin >> c;
Switch(c)
{Case '#':
cout << " Enter radious \t" << endl;
cin >> r;
x = (3.14 * r * r);
cout << "This is area of circle \t" << x << endl;
y = (2 * Pi * r);
cout << " This is circumfrance of the circle \t" << y << endl;
break;
Case '*':
        cout << " Enter the first value \t " << endl;
        cin >> a;
        cout << " Enter the 2nd value \t" << endl;
        cin >> b;
        z = a + b;
        cout << " Result of addition \t" << z << endl;
        z = a / b ;
        Cout << "Result of division \t" << z << endl;
        break;
        default:
        Cout << "invalid No." << endl; }
        getchar();
        getchar(); }
```

Program write a program that display the following out put on the consoul.

Q NO3 out put 0, 2, 4, 6, 8 Hint (use for loop) 70

≡

```
{int a,b;
cout<< "Enter the no." <<endl;
cin >>b;
for(a=0; a<b; a+=2)
 cout << a <<endl;
 getchar();
 getchar();
```

(2) while loop

Syntax.

```
while (condition)
  {statements;}
```

one or multibles statements

(3) do while loop.

```
do
  {statements;}
while (condition);
```

71

① For loop	output	② while loop	output	③ dowhile loop
≡	1	≡	1	≡
{ int K;	2	{ int K=1	2	{ int K=1
for(K=1; K<6; K++)	3	while (K<6)	3	do
{ cout << K << endl;}	4	{ cout<<K<<endl;	4	{ cout << K<<endl;
system("Pause");	5	K++; }	5	K++; }
}		system("Pause"); }		while(K<6);
				system("Pause");
				}

output (③): 1, 2, 3, 4, 5

"نوٹ" جب ہم K کی value (-ve) رکھیں گے تو یہ ساری کام نہیں کریں گا سوائے
(3) loop کے وہ صرف ایک output ضرور شو کروائے گا۔
یعنی اس میں ایک بار statement ضروری execute ہو گا چاہے
Statements کی false ہی کیوں نہ ہو۔

---

① Continue Statement جاری رہنا
② Break Statement.
③ go to Statement.

Program.

① Continue Statement. ② & break Statement.

```
≡
{ for (int K=1 ; K<6 ; K++)
{ cout << " C++ " << endl;
continue; (OR) break;
Cout << " JAVA " <<endl; }
Cout << " Good thinking " <<endl;
system ( " Pause " ); }
```

72

Note:

continue اور break کو ہم body of loop میں استعمال کر سکتے ہیں۔
ہم ایک پروگرام میں break اور continue کو آگے پیچھے بھی استعمال کر سکتے ہیں۔

goto کو ہم جہاں دل چاہے استعمال کر سکتے ہیں۔ اس کو ہم loop کے علاوہ بھی استعمال کر سکتے ہیں۔

Program

```
{ label:
   cout << " Good Class" <<endl;
   Cout << " Bad class" <<endl;
   cout<< " Hot day" <<endl;
   goto label;
   System ("Pause"); }
```

⇒ Nested loop. Syntax.

```
for ( initialization; Condition ; Inc/ dec )
      { for ( initialization; condition; Inc/ dec)
            { Statements ; }
      }
```

Similarly Nested while loop & Nested do while loop

73

Row ← outer loop کے لیے

colom ← iner loop کے لیے

جب outer کی condition ← True ہو جاتی ہے تو inner loop کی طرف جاتا ہے گا اور جب inner کی condition false ہو جاتی ہے تو Compiler outer کی طرف جاتا ہے گا۔

Program =

```
{ for (int x=1 ; x<5 ; x++)
   { for (int y=2; y<5; y++)
      { cout << " * "; }
      cout << endl; }
   system("Pause");
}
```

Dry Run

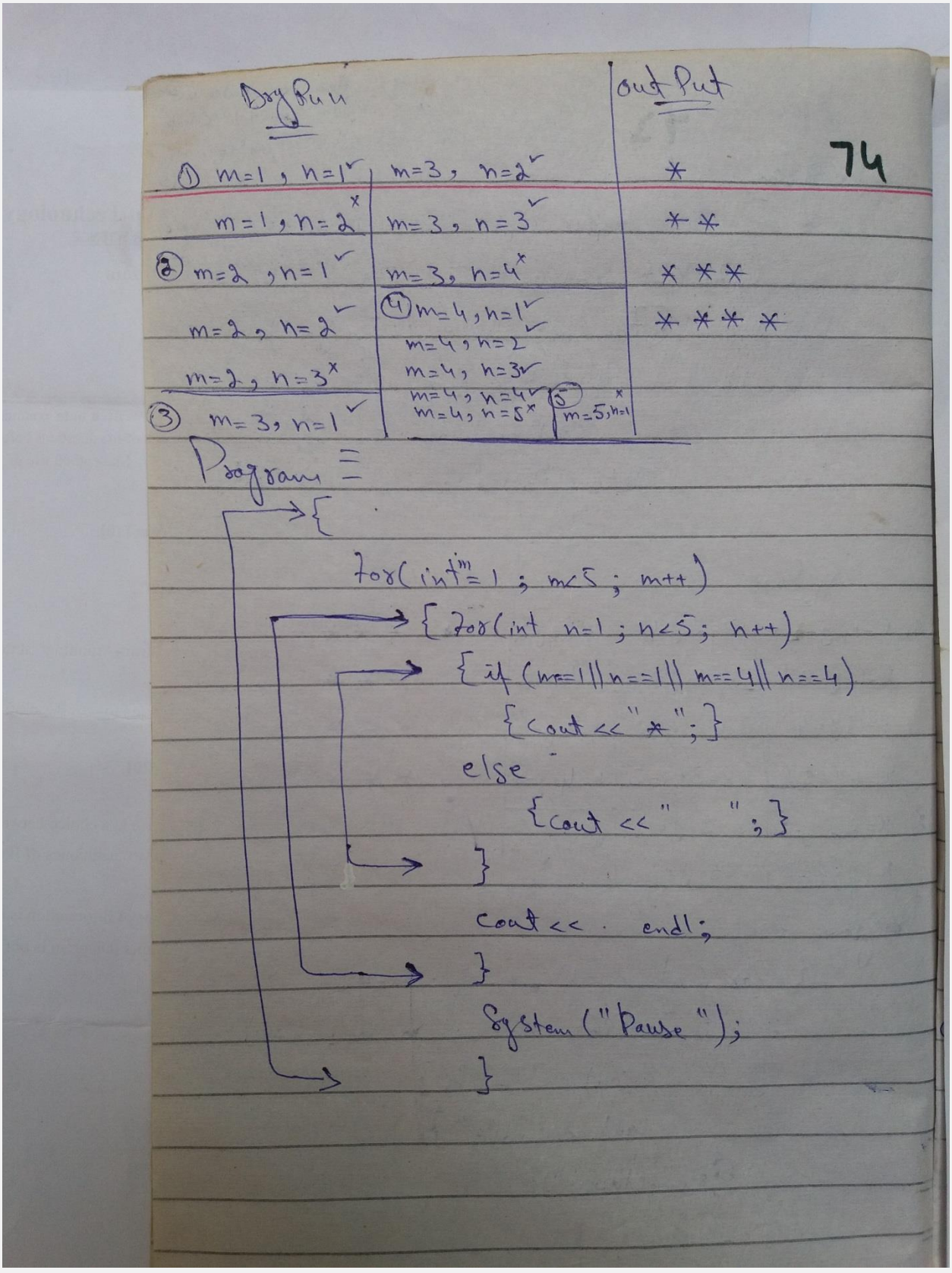
① x=1, y=2 ✓	x=2, y=4 ✓	x=4, y=2 ✓
x=1, y=3 ✓	x=2, y=5 ✗	x=4, y=3 ✓
x=1, y=4 ✓	③ x=3, y=2 ✓	x=4, y=4 ✓
x=1, y=5 ✗	x=3, y=3 ✓	x=4, y=5 ✗
② x=2, y=2 ✓	x=3, y=4 ✓	x=5, y=5 ✗
x=2, y=3 ✓	x=3, y=5 ✗	

out put

```
* * *
* * *
* * *
* * *
```

Program =

```
{ for (int m=1 ; m<5 ; m++)
   { for(int n=1 ; n<=m ; n++)
      {cout << " * ";}
      cout << endl; }
   System("Pause");
}
```



Dry Run

		Out Put
① m=1, n=1 ✓	m=3, n=2 ✓	*
m=1, n=2 ✗	m=3, n=3 ✓	* *
② m=2, n=1 ✓	m=3, n=4 ✗	* * *
m=2, n=2 ✓	④ m=4, n=1 ✓	* * * *
m=2, n=3 ✗	m=4, n=2 ✓	
③ m=3, n=1 ✓	m=4, n=3 ✓	
	m=4, n=4 ✓	
	m=4, n=5 ✗	⑤ m=5, n=1 ✗

74

Program =

```
{
for(int m=1; m<5; m++)
{ for(int n=1; n<5; n++)
{ if (m==1||n==1|| m==4|| n==4)
{cout << "*";}
else
{cout << " ";}
}
cout << endl;
}
System("Pause");
}
```

Dry Run

			Out Put
m=1, n=1 ✓	m=2, n=2 ✗	m=3, n=4 ✓	* * * *
m=1, n=2 ✓	m=2, n=3 ✗	m=4, n=1 ✓	* . . *
m=1, n=3 ✓	m=2, n=4 ✓	m=4, n=2 ✓	
m=1, n=4 ✓	m=3, n=1 ✓	m=4, n=3 ✓	* . . *
m=1, n=5 ✗	m=3, n=2 ✗	m=4, n=4	* * * *
m=2, n=1 ✓	m=3, n=3 ✗	m=5, n=1 ✗	
		-	

75

Funtion ch#5

Funtions are two types.

① user define funtion. eg ① add() ② show() ③ disply()

② Built in funtion. eg ① sin() ② cos()

Three tooms are used in funtion.

① Defining the funtion.

② Declaring the funtion

③ calling the funtion.

① Defining the funtion.

Syntax

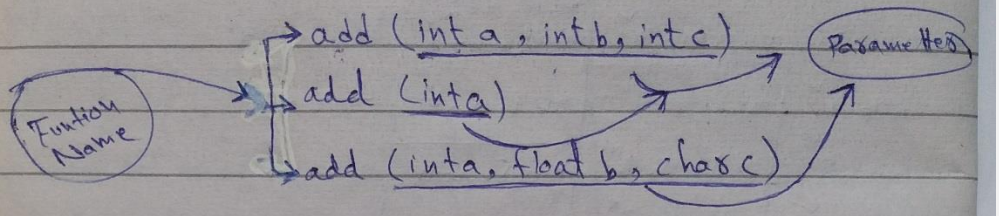
Return type _ funtion Name (Parameters)

↙ space

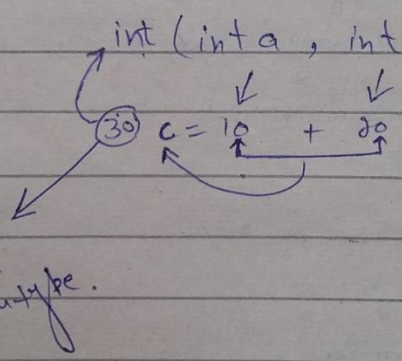
{ Statements ; }

76

جب ہم کسی function کے اندر کوئی variable یا Data type لکھتے ہیں تو وہ Parameter کہلاتے ہیں۔ مثلاً



Datatype وہ جواب ہوگا وہ ہی اس فنکشن کی final ویلیو کا ہوگا۔ Returntype کہلاتی ہے اور eg.



جب ہم datatype کی جگہ void (Reyword) لکھتے ہیں تو اس کا مطلب ہوتا ہے کہ function کی Returntype نہ ہو۔ eg. void

Void show( ) OR Void Show(void)

No Datatype → No return type.

No Parameters

eg int factorial (int n)
{ ≡ }

٦٧

② Declaring the funtion.

Syntax

Returntype ␣ funtionName(Paramettes);
(space)

③ Calling the funtion.

Syntax

funtionName (Aregguments);

اگر ہم simple a, b, c کو دیتے ہیں تو وہ Aregguments کہلاتے ہیں
یعنی ان کی Datatype کے بغیر دیتے ہیں تو وہ Aregguments کہلاتے ہیں
وہ ایک دو یا تین بھی ہو سکتے ہیں۔

eg add (x, y);

Parametter اور Aregguments کے variables ایک جیسے same نہیں ہونے چاہئیں۔

show ( );
factorial(x);

⇒ Two way of defining the funtion.
① Befor the main() funtion.
② After the main() funtion.

جب ہم

Note:
when we define the funtion Before the main() funtion. There is no need of declartion of funtion. But when we

78

define the funtion after the main funtion, funtion declartion must be written.

Returntype funtionName (Parameter)
{ Statement; }

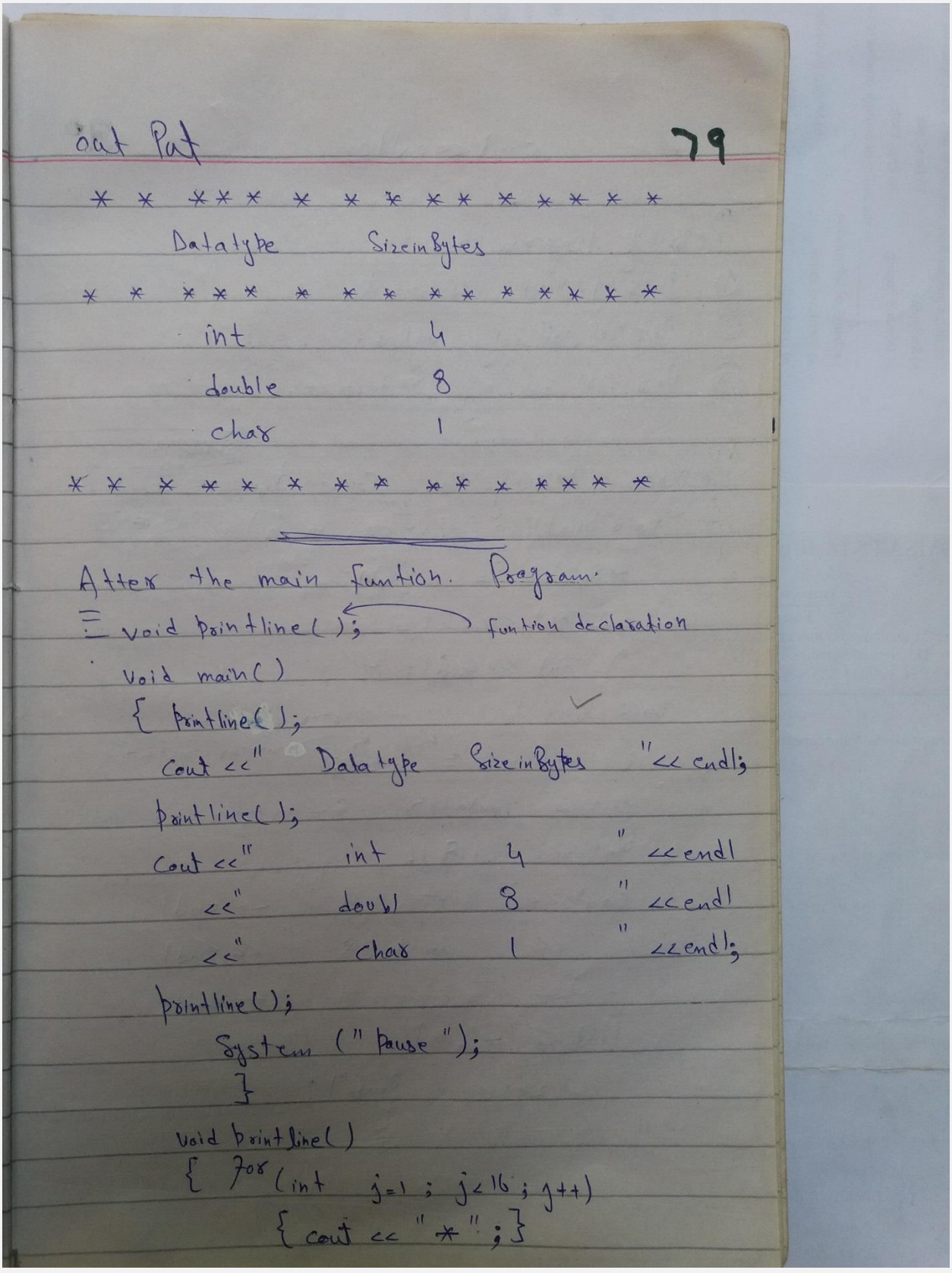
Funtion Header

Body of funtion

Program. (Befor the main funtion)

```
Void Printline()
  { for(int j=1; j<=16; j++)
      { cout << "*"; }
      cout << endl; }

Void main()
 { Print line();
   cout << "    Datatype    Size in Bytes  " << endl;
 Print line();
 cout << "    int         4              " << endl
      << "    double      8              " << endl
      << "    Char        1              " << endl;
 Printline();
        System ("Pause");
    }
```



out Put 79

* * *** * * * * * * * * * *

Datatype Size in Bytes

* * *** * * * * * * * * * *

int 4

double 8

char 1

* * * * * * * * ** * *** *

After the main funtion. Program.

= void printline( ); funtion declaration

void main( )

{ printline( );

cout << " Datatype Size in Bytes " << endl;

printline( );

cout << " int 4 " << endl

<< " doubl 8 " << endl

<< " char 1 " << endl;

printline( );

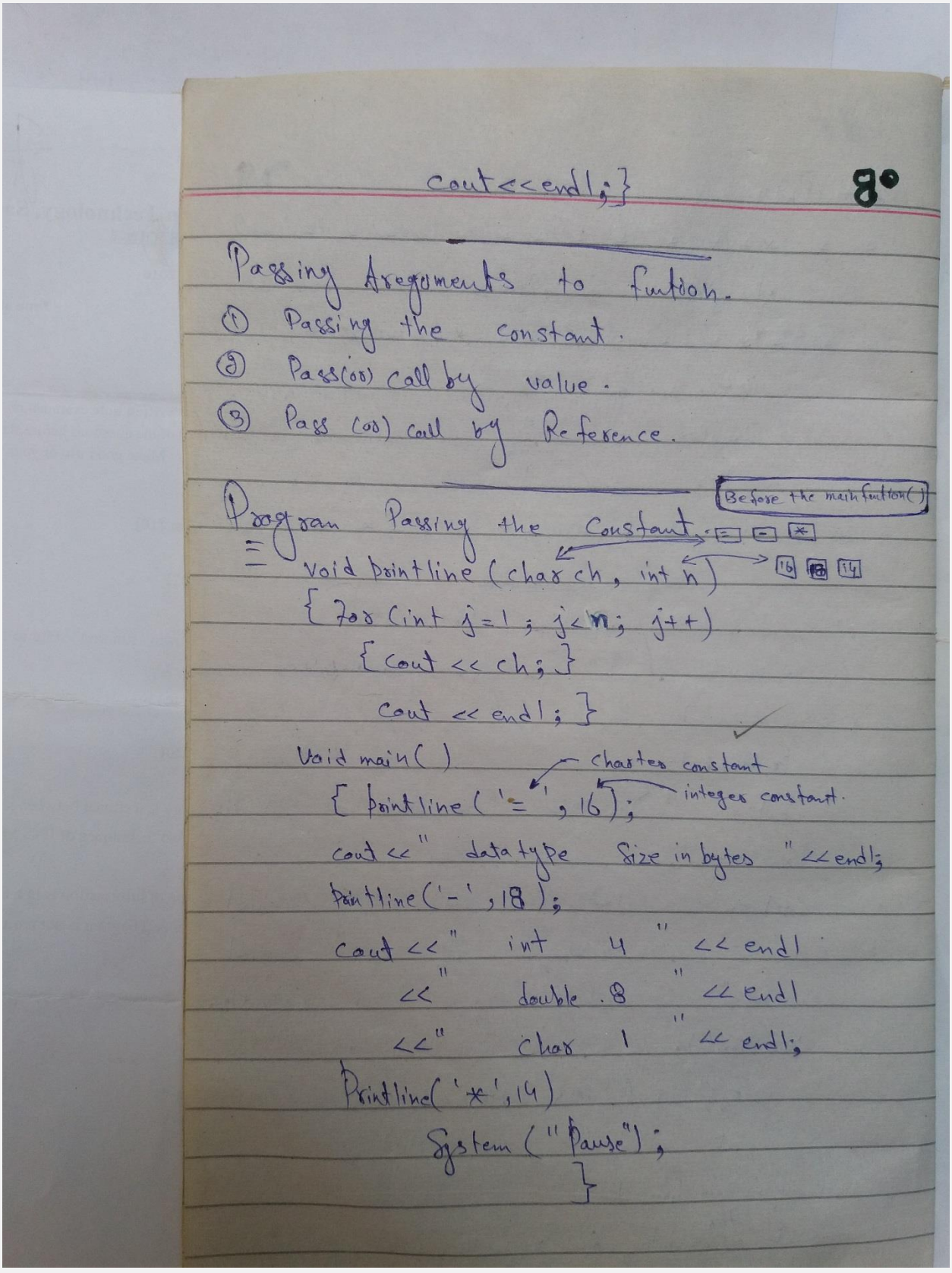
System (" pause ");

}

void printline( )

{ for (int j=1 ; j<16 ; j++)

{ cout << " * " ; }



cout<<endl;}

Passing Aregements to futions.

① Passing the constant.

② Pass(oo) call by value.

③ Pass (oo) call by Reference.

Before the main function()

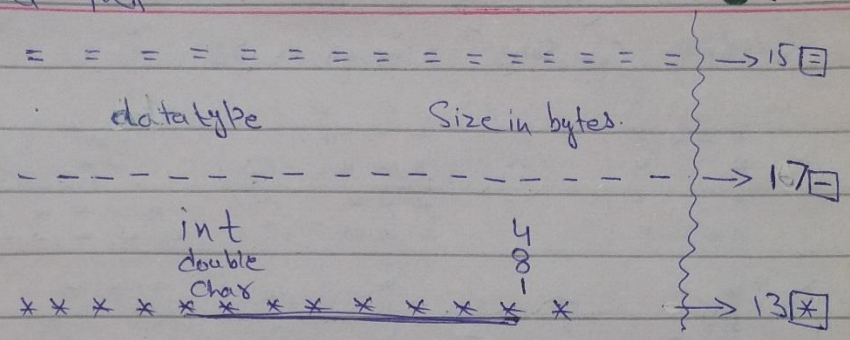
① Program Passing the Constant.

```
void printline (char ch, int n)
{ for (int j=1 ; j<n; j++)
  {cout << ch;}
   cout << endl; }
Void main( )
{ printline ('=', 16);
cout << "  datatype   Size in bytes " <<endl;
printline('-',18);
cout <<"   int      4    " << endl
     <<"   double   8    " <<endl
     <<"   char     1    " << endl;
Printline('*',14)
    System ("Pause");
        }
```

charter constant

integer constant.

out put 81



No1 write a program that print three lines of stars using forloop in the user defined funtion named as printline(). You also call call printline() funtion from the main() funtion according to your need & also shows size of data types using simple cout objed in main funtion.

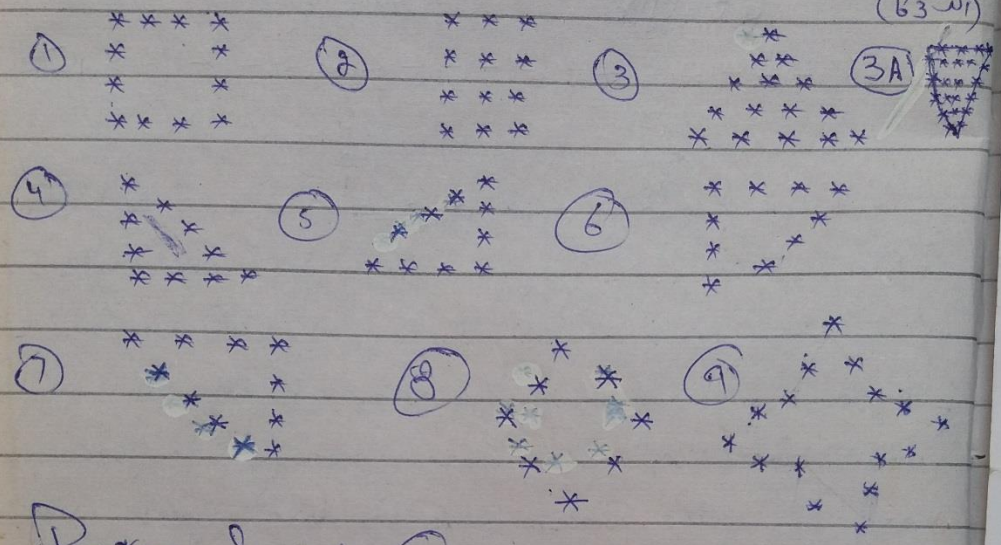
Program.

```
#include<iostream>
using name space std;
Void printline();
{for(int j=1; j<16; j++)
  {cout<<"*";};
  cout<<endl;}

Void main()
{ printline();
  cout<<"  Datatype  size in bytes  "<<endl;
printline();
  cout<<"  double    8              "<<endl
  <<"      int       4              "<<endl
  <<"      char      1              "<<endl;
printline();
        getchar();
        getchar();}
```

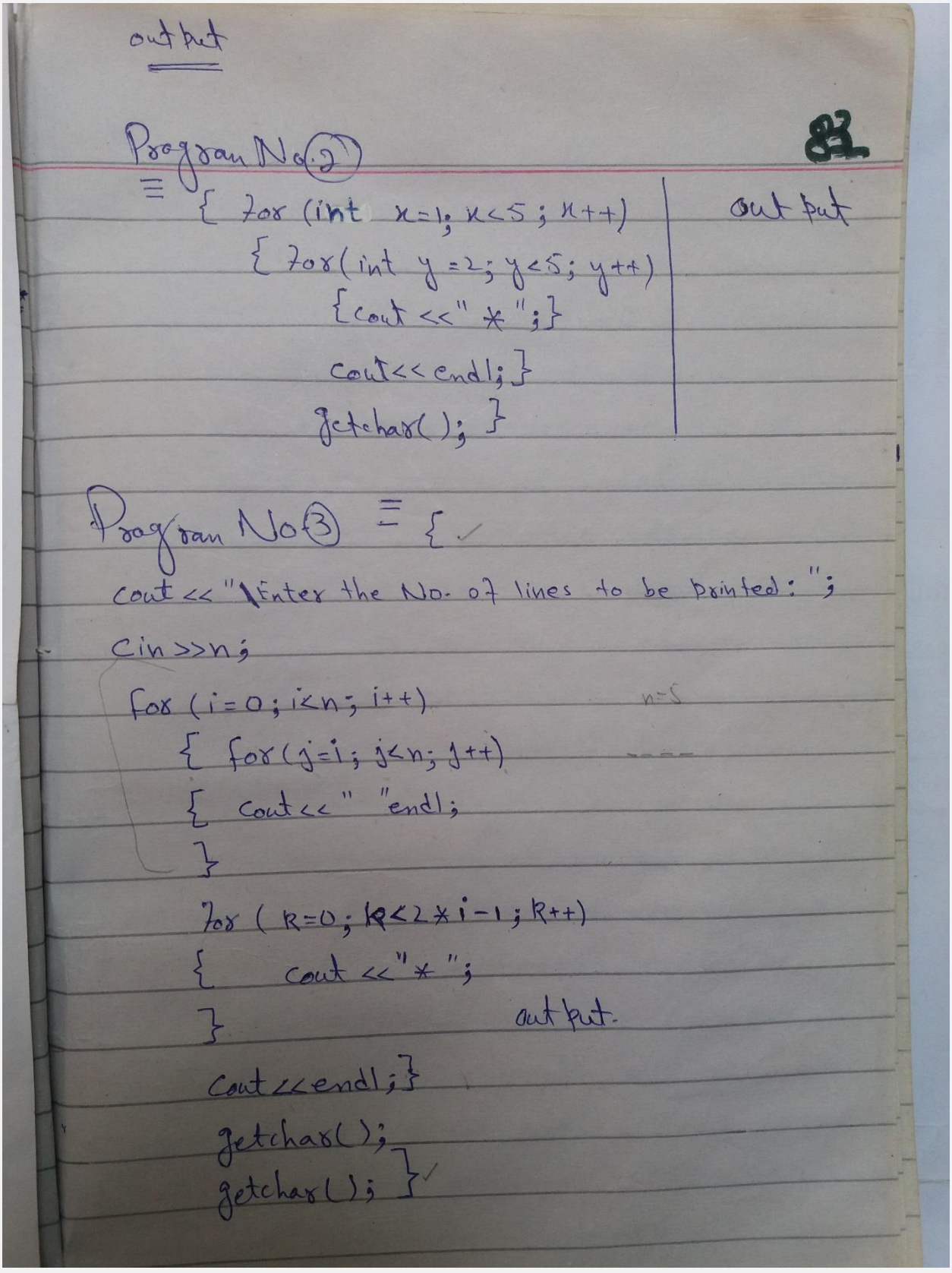
Program.

Write a Program display the following output 82
on the Console (Nested loop)



Program No. 1

```
{ for(int m=1; m<5; m++)
  { for(int n=1; n<5; n++)
    {if (m==1||n==1||m==4||n==4)
      { cout<<"*";}
    else
      { cout<<" ";}
    }
  cout << endl;
  }
    getch();
}
```



output

Program No (2)

```
{ for (int x=1; x<5; x++)
    { for(int y=2; y<5; y++)
        {cout << " * ";}
      cout<<endl;}
    getchar(); }
```

output

Program No (3) = {

```
cout << "Enter the No. of lines to be printed: ";
cin >>n;
for (i=0; i<n; i++)                    n=5
   { for(j=i; j<n; j++)
   { cout<<" "endl;
   }
   for ( k=0; k<2*i-1; k++)
   {    cout << "* ";
   }                                   output.
   cout<<endl;}
   getchar();
   getchar(); }
```

Passing Arguments to Functions. 84

(2) Pass/call by value. ⎤ defination on
(3) Pass/call by Reference. ⎦ Next Page.

=> Reference operater. (or) Address operter (&)

کسی ویری ایبل کا Memory location یا address یا cell کو show کروانا ہو تو ہم
Program میں Reference opr/Add op استعمال کرتے ہیں۔

```
≡ { int z=10;
    cout << "The value of z" << z endl;
    cout << "The address of z" << &z << endl;
    getchar();
    getchar(); }
```

(2) Pass/call by value. Program.

```
# ≡
                              formal Parameter
void duplicate(int a, int b)
   * calledfunction.          -> scope.
      { a*=2;   /  a=a*2;
        b*=4;   /  b=b*4;
   cout << a << endl << b << endl; }
void main()
  {  int x, y;
  Cout << "Enter the value for x" << endl;
  cin >> x;
```

85

```
cout << " Enter the value for y " << endl;
cin >> y;
duplicate (x, y);
```

Actual Parameters (OR) actual Arguments.

* Calling function

```
cout << x << endl << y << endl;
        getchar();
        getchar();
```

Definition:

A procedure (طریقہ) in which we coby the values of actual parameters to the formal parameter is called call/Pass by value.

A procedure in which the address of actual parameter goes to formal parameter is called by call/Pass Reference.

Program.

Seme Program as above but little difference

=> void duplicate (int &a, int &b)

(out Put)

Pass by value

Enter the value for x 2

Enter the value for y 3

4
12
2
3

Pass by Reference

-> 2

-> 3

4
12
4
12

⇒ Types of variables according to function. 86

There are (04) Variable.

(1) Local variable (scope + life time)
(2) Global variable (scope + life time)
(3) Static variable (scope + life time)
(4) Register variable (X + X)

(1) Local variable.

⇒ A variable declared inside a function is known as local variable (or) variable that is return inside the user define function is called Local variable.

⇒ The area / place where the variable can be accessed or written is called scope of variable.

⇒ The time period by which the variable remain in computer memory.

So,
The Scope of Local variable inside the user define function. (a,b)
& Life time is also inside the user define function.

(a,b) کی values ترلیز سے باہر آتے ہی
delete ہو جائے گی،

87

② Global variable

Global variable is the variable that is written out side of any funtion.
Scope remain untill end of the program
Life time is also untill end of the program.
یہ پروگرام کے آخر تک Memory میں رہتا ہے اور Globaly سے access ہے۔

=

```
int z;   → z is Global variable
void show()
{ z = z+3; }
void main()
{ cout << "Enter the value";
  cin >> z;
  Show();
  Cout << z << endl;
      getchar();
      getchar(); }
```

③ Static variable.

A local variable that is return inside user define function with static key word is called static variable.

void duplicate (static int a, static int b)

Scope is Similarly to the scope of Local variable & Life time is same as Global variable. 68

(4) Register variable

A variable that is return with register Key variable (or) in register variable the values of variables are stored in Register instead of RAM, Beause register are faster than RAM & slow than cash memory.

void duplicate (register int a; register int b)

Program.

Funtion overloading function Name Same But signatures are different.

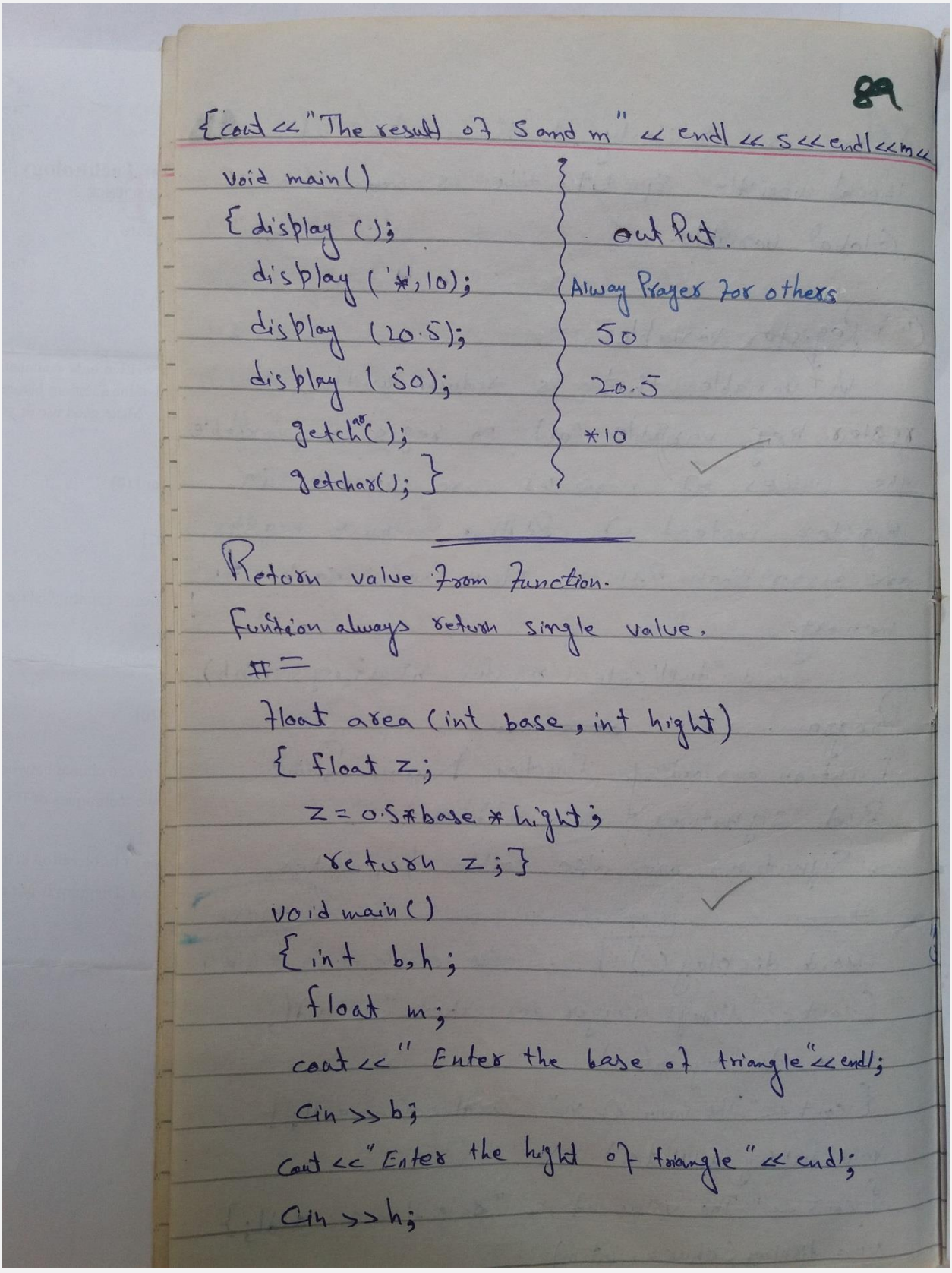
=> Signatures are also called parameter.

#

Note

Float استعمال نہیں کرتے double استعمال کرتے ہیں

void display ( )
{cout<<" Always Prayer for others "<< endl;
void display(int n)
{ cout<<" The value of n "<<endl<<n<<endl;}
void display (double c)
{ cout<<" The value of c "<<endl<<c<<endl;}
void display (char s, int m)



89

{cout << "The result of 5 and m" << endl << 5 << endl << m <<

```
void main()
{ display ();
  display ('*',10);
  display (20.5);
  display (50);
    getch();
    getchar(); }
```

output.

Alway Prayer for others
50
20.5
*10

Return value from function.

Function always return single value.

#=

```
float area (int base, int hight)
{ float z;
  z = 0.5*base*hight;
   return z;}
void main()
{ int b,h;
  float m;
 cout << " Enter the base of triangle" << endl;
 Cin >> b;
 Cout << "Enter the hight of triangle " << endl;
 Cin >> h;
```

endl;}

90

```
m = area(b,h);
cout <<" The area of triangle"<<m<<endl;
        getchar();
        getchar(); }
```

Area = $\frac{1}{2}$ * base * hight.

Return value کی value فنکشن سے مین میں آتی ہے۔

value/char کی Return ہو سکتی ہے۔ مگر فنکشن صرف ایک value کو

کی return کرواتا ہے۔

return 10; (OR) return A;

Program.

```
#include <iostream>
using namespace std;
int main()
{cout<<" Best of Luck!"<<endl;
return 0;
}
```

Cout put result.

Best of Luck!

Arrays ترتیب - سنوارنا attached 91

It is an arrged set of locations. any of which can be accessed from some common starting address.

(OR)

(one after other)

Consecutive (↑) memory location with same name & same data type.

3 terms in array.
- memory consective location
- same data type

Genaric = عام چیز ← ms word

specific = خاص چیز ← IDE

4 things in arrays.

(1) name of Arrays

(2) Elements of array

(3) Lenghth of array.

(4) index of Array.

① اصل میں variable user define کو Name of Arrays کہتے ہیں۔

marks | 10 | -2 | 10 | 50 | (index 0 … 3)

② Array میں جتنی values ہوتی ہیں وہ Array کے elements ہیں۔ جیسے اوپر 4 element ہیں 5 یا 6 بھی ہو سکتے ہیں۔

③

Total Number of elements in Array is called its length. eg (4) length of above example

④

Index is basically memory location / Address In the array index is always start from zero.

it refers to the number that identifies a specific element in an array.

First index = 0 92

Find last index = Length − 1
= 4 − 1 = 3

Syntax Declaration.

Datatype Name of Array [length of Array]; (sequire BracKt)

```
int marks [4];
marks [0] = 10;
marks [1] = -2;
marks [2] = 10;
marks [3] = 50;
```

Array کا مقصد values کو serch کرنا Assending یا decsending ترتیب کو معلوم کرنا. Maximum یا Minimum کو معلوم کرنا

Syntax. Intialization

Datatype Name of Array [length] = {List of values};

```
int marks [4] = {10, -2, 10, 50};
```

⇒ Note
cin کے ساتھ نہیں ہے. endl; کو استعمال نہیں کر سکتے - صرف cout کے ساتھ ہم استعمال کر سکتے

Program Array declaration. 93

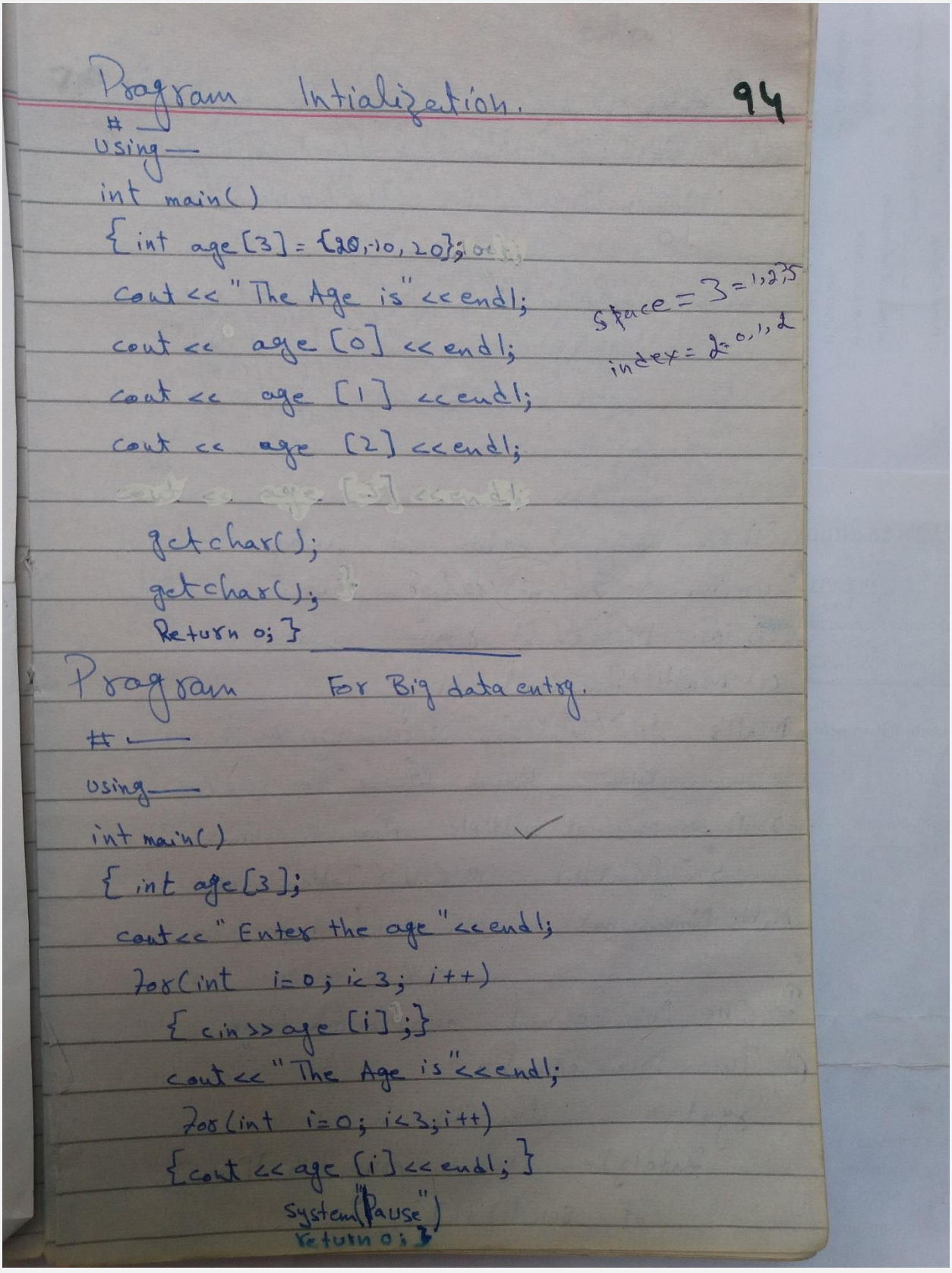
```
# ____

using ____
int main ( )
{ int age [3];   <- Declare
                 <- index (0-2)
cout<<" Enter the Age "<<endl;
cin>> age [0];  }
cin >> age [1]; } OR cin>>age[0]>>age[1]>>age[2];
cin >> age [2]; }
cout<<"The Age is"<<endl;
cout<<age [0] <<endl; }
cout<<age [1]<< endl; } cout<<age[0]<<age[1]<<age[2];
cout<<age [2]<<endl;  }
        getchar();
        getchar(); }
```

Dry run	output.	
age [0] =	Enter the Age	The Age is
age [1] =	15	15
	20	20
age [2] =	100	100

```
int a;
cout<<"Enter Number"<<endl;
cin>>a;
getchar(); }
```

Enter No
(10)



Program Intialization. 94

#
Using—

```
int main()
{ int age[3] = {20,-10,20};
cout << "The Age is" << endl;
cout << age[0] << endl;
cout << age[1] << endl;
cout << age[2] << endl;
getchar();
getchar();
Return 0; }
```

space = 3 = 1,2,3
index = 3 = 0,1,2

Program For Big data entry.

#
Using—

```
int main()
{ int age[3];
cout << "Enter the age" << endl;
for(int i=0; i<3; i++)
{ cin >> age[i];}
cout << "The Age is" << endl;
for(int i=0; i<3; i++)
{cout << age[i] << endl; }
System("Pause")
return 0; }
```

out Put

Enter the Age — The Age is — 95

Enter the Age	The Age is
50	50
20	20
10	10

as age[i] ← age[0], age[1], age[2]

There are 3 type of Arrays

(1) One Dimensional Array. 1-D

(2) Two Dimensional Array 2-D

(3) Multi Dimensional Array

Details 1-D اس میں Row اور Colum بھی اکٹھے نہیں آتے تھا۔
دونوں میں سے کوئی ایک آتا یا Row یا Colum

2-D ⇒ one or multiple Row & Colums
مثلاً Table کی فارم میں output آئے گی۔

Multi Dimensional اس میں بہت سی ٹیبل ہوتے ہیں۔
یعنی matrix بناتے ہیں۔

① ⇒ One Dimensional on Previous Page

② Two Dimensional.

Syntax Declared.

Datatype Name of Array [Row][Colum];

int sun [3][2];

Syntax Initialized

96

Datatype Array Name [Row][colum] = {list of values};

int sum [3][3] = {2,4,5};

	0	1	2
0	(0,0) 10	(0,1) 20	(0,2) 30
1	(1,0) 40	(1,1) 50	(1,2) 60
2	(2,0) 70	(2,1) 80	(2,2) 90

→ index of colum

OR

index of Row

[0][0] = 10
[0][1] = 20
[0][2] = 30
[1][0] = 40
[1][1] = 50
[1][2] = 60
[2][0] = 70
[2][1] = 80
[2][2] = 90

Program of two-Dimensional. ۹۷

1-D کے اندر ایک loop
2-D کے اندر دو loops اور multi D کے اندر Multiloops ہوتی ہے۔
const کا مطلب value کو fix کرنا ہے۔
const کو ہم مین کے اندر نہیں لکھتے
const اصل میں constant ہے۔
const is Keyword.

```
# ______
using ______
const int District = 3;
const int Month = 2;
void main()
{double S[3][2];  for(int d=0; d<District; d++)
  { for(int m=0; m<month; m++)
      {cout<<"Enter the Sales for District "<<d+1;
      cout<<"AND Month"<<m+1<<":";}
      cin >> S[d][m];}
  }
  cout << "\n\n";
  cout << " months " <<endl;
  cout << "_1_ _ _ _2_ _ _ _ _ _ _"<< endl;
  cout << "   1   2  " << endl;
  cout << " _ _ _ _ _ _ _ _ _ _ _ "<<endl;
  {for(int d=0; d<District; d++)
     { for( int m=0; m< month; m++)
     {cout << "\t "<< S[d][m]<<"\t";}
     cout <<endl; }
  }

     getchar();
     getchar(); }
```

Out Put

Enter the Sales for District 1 AND Month 1: 20
Enter the Sales for District 1 AND Month 2: 30
false ⇒
Enter the Sales for District 2 AND Month 1: 15
Enter the Sales for District 2 AND Month 2: 25
false ⇒
Enter the Sales for District 3 AND Month 1: 9.50
Enter the Sale for District 3 AND Month 2: 10
false ⇒

months	
1	2
20	30
15	25
9.50	10

Dry Run 98

s[0][0]	d=0, m=0
s[0][1]	d=0, m=1
s[0][2]	d=0, m=2
s[1][0]	d=1, m=0
s[1][1]	d=1, m=1
s[1][2]	d=1, m=2
s[2][0]	d=2, m=0
s[2][1]	d=2, m=1
s[2][2]	d=2, m=2

d=0, m=0
d=0, m=1
d=1 m=0
d=1 m=1
d=2 m=0
d=2 m=1

initalized

Datatyp Name of Arry [][] = { List of value };

Same out Put.

int s[3][2] = {20,30,15,25,9.50,10};

Searcing in Array.

99

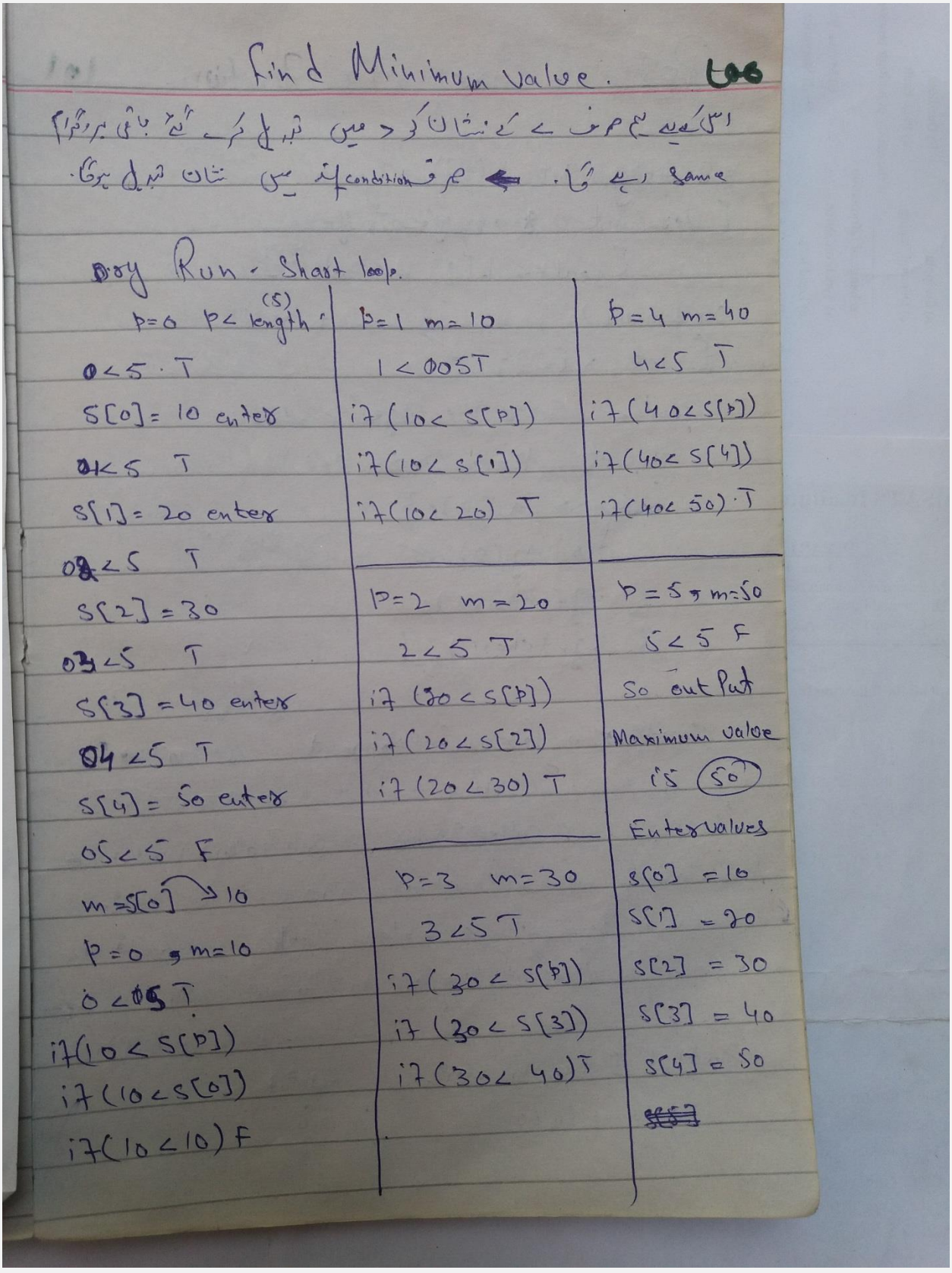
#

using

Find Maximum value

```
Const int length = 5;
Void main()                    index (0-4)
 { int s[Length], m;
   cout << "Enter the Elements" << endl;
   For (int p=0; p<length; p++)
     { cin >> s[p];}
   m= s[0];
   for (int p=0; p<length ; p++)
   {if (m< s[p])        for mini
      { m= s[p];}
   }
   Cout << "Maximum value i's" << m<<endl;
                getchar();
                getchar(); }
```

output کے بعد Rule کے مطابق Row نارمل میں شو ہونا چاہیے مگر property کی 1-DArray کے values شو کروائے گا یعنی endl; کے بغیر وہ نارمل میں values کو شو کروائے گا۔



Find Minimum value.

اس کوڈ میں صرف کے نشان کو < میں تبدیل کرنے باقی برقرار
رہے گا۔ ⬅ صرف condition میں نشان تبدیل ہوگا۔

Dry Run - Short loop.

p=0 p< length (5)

0<5 T
S[0] = 10 enter
1<5 T
S[1] = 20 enter
2<5 T
S[2] = 30
3<5 T
S[3] = 40 enter
4<5 T
S[4] = 50 enter
5<5 F
m = S[0] → 10
p=0 , m=10
0<5 T
if(10 < S[p])
if(10 < S[0])
if(10 < 10) F

p=1 m=10
1<5 T
if(10 < S[p])
if(10 < S[1])
if(10 < 20) T

p=2 m=20
2<5 T
if(30 < S[p])
if(20 < S[2])
if(20 < 30) T

p=3 m=30
3<5 T
if(30 < S[p])
if(30 < S[3])
if(30 < 40) T

p=4 m=40
4<5 T
if(40 < S[p])
if(40 < S[4])
if(40 < 50) T

p=5 , m=50
5<5 F
So out Put
Maximum value is (50)

Enter values
S[0] = 10
S[1] = 20
S[2] = 30
S[3] = 40
S[4] = 50

Passing Array to Function. 101

=

```
Void display (int b[5])        <- Called Function
{ for (int g=0; g<5; g++)
      { cout << b[g] <<endl; } }
Void main( )
{ int a[5];                    <- Datatype
  cout << "Enter the Age" <<endl;
  for(int p=0, p<5; p++)
  { cin>> a[p]; }
      display (a);             <- Calling function
      getchar();
      getchar(); }
```

calling function میں جو Argument (Parameter) استعمال ہوگا وہ ہی فنکشن میں Name of variable کے طور پر استعمال ہوگا۔

← calling function میں اور called function میں loop کے variable بھی مختلف ہوں گے مگر اگر فنکشن کو called میں کیا تو دونوں loop کے variables ایک جیسے ہو سکتے ہیں

← called فنکشن میں length لکھے گئے مگر calling میں length نہیں لکھتے۔

OutPut Enter the Age.

20 30 40 50 60

out Put

20
30
40
50
60

Multi Dimensional Array.

Syntax

Data type Name of Array [Length1][length2][Length3]...[lengthn];

int a[2][3][4];

Exp=> int b[2][3][4];

```
for(int i=0; i<2; i++)
{ for (int j=0; j<3; j++)
  { for(int k=0; k<4; k++)
    { cin>> b[i][j][k]; }
  }
}
```

(1) (2) (3)

Built-in-function / Math function.

(1) abs() → Absolute funtion -1 = 1

(2) fabs() → Fractional Absolute funtion -0.1 = 0.1

(3) ceil() 0.5 = 1 , -0.5 = -0

(4) floor() 0.5 = 0 , -0.5 = -1

(5) Sin()

(6) Cos()

(7) tan()

(8) log()

(9) log10()

(10) sqrt()

103

(11) pow (a,b) → 2nd Parameter always for exponent

Note pow(2,3) is mean $2^3 = 8$

① صرف absolute فنکشن کے ساتھ int آئے گا باقیوں کے ساتھ نہیں آئے گا۔ یعنی absolute نہ Float ہوتا ہے اور نہ ہی char صرف اور صرف int data type ہو گی۔

② pow کے علاوہ باقی سبھی میں ایک ہی parameter آئے گا۔

③ pow() میں صرف ② variable آئیں گے مگر باقی ساروں کے ساتھ صرف ایک variable یعنی Parameter | Argument آئے گا۔

Function calling Rule.

User define function. (UDF)

✓ UDF میں ہم values کو cout کرواتے ہیں۔ اگر return نہ کیا ہو

UDF میں ہم constand کو Pass کرواتے ہیں۔

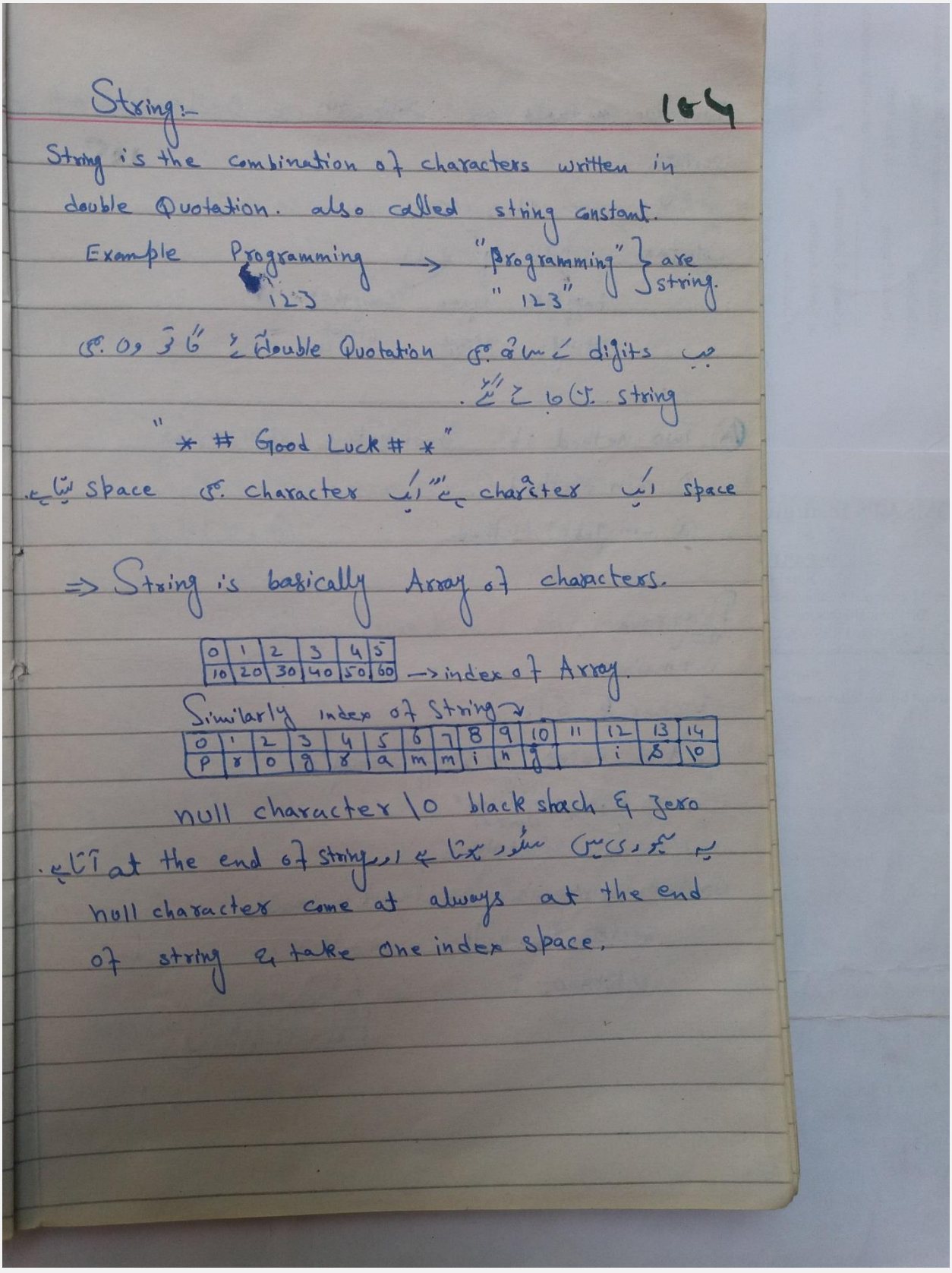
✓ UDF میں ہم سارے ہی loops, if else conditions, switch آئے گا۔

Main function. (MF)

✓ MF میں ہم calling کرتے ہیں۔

✓ MF میں ہم values کو cin کرواتے ہیں۔

✓ UDF میں return دے گے تو cout بھی main میں آئے گا۔ کیونکہ value Return ہو کر main کے variable میں سٹور ہوتی ہے۔



String:- 104

String is the combination of characters written in double Quotation. also called string constant.

Example Programming → "Programming" } are string.
123 "123"

جب digits کو double Quotation میں لکھا جائے تو وہ string بن جاتے ہیں۔

"* # Good Luck # *"

space بھی character ہے character ایک space ہوتا ہے۔

⇒ String is basically Array of characters.

0	1	2	3	4	5
10	20	30	40	50	60

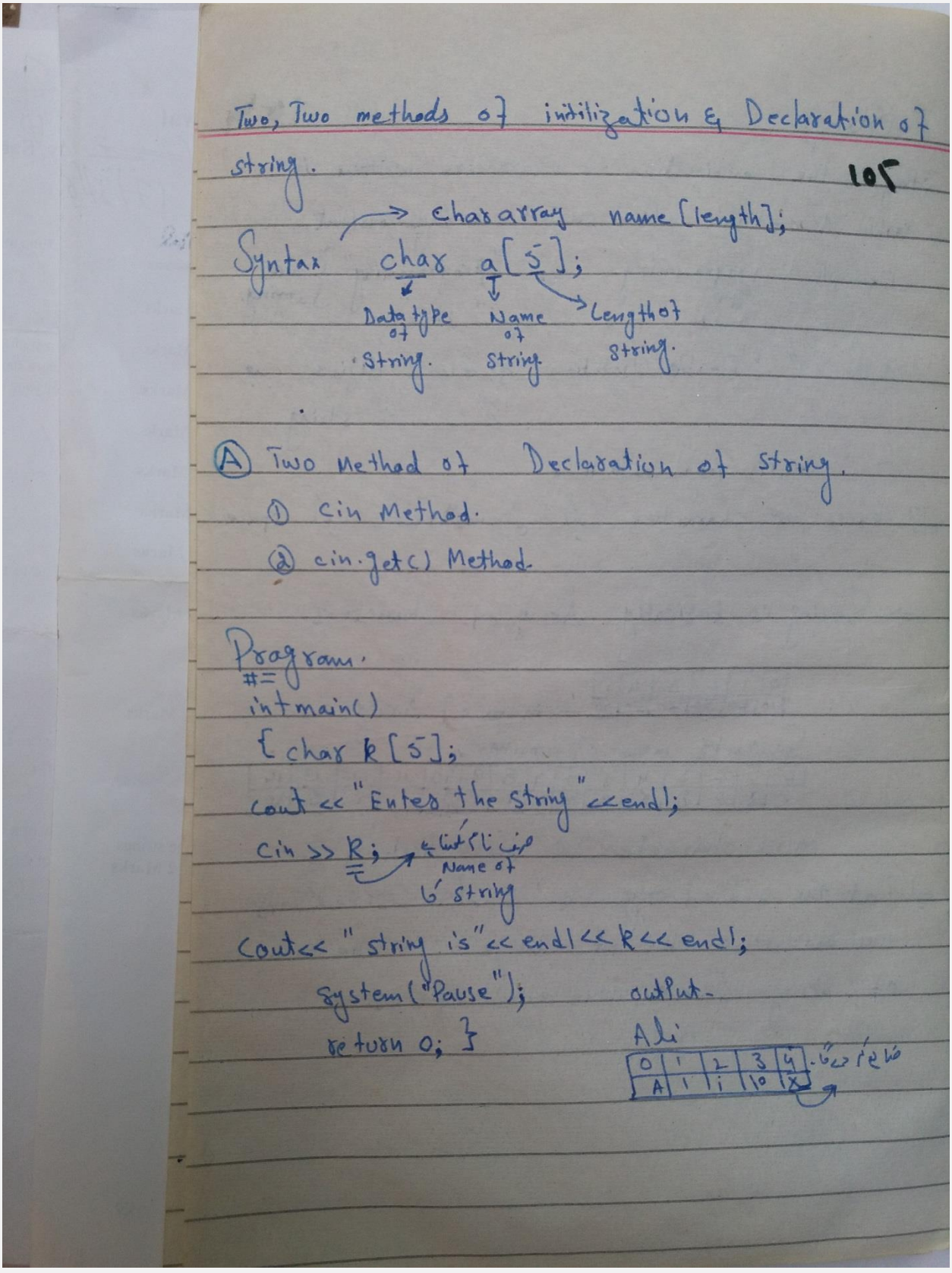
→ index of Array.

Similarly index of String →

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
P	r	o	g	r	a	m	m	i	n	g		i	s	10

null character \0 black slash & zero یہ ہمیشہ string کے آخر میں آتا ہے۔ at the end of string

null character come at always at the end of string & take one index space.



Two, Two methods of initilization & Declaration of string.

105

Syntax → char array name [length];

char a[5];

- char → Data type of String.
- a → Name of String.
- 5 → Length of String.

(A) Two Method of Declaration of String.

① cin Method.

② cin.get() Method.

Program.

```
#=
int main()
{ char k[5];
cout << "Enter the string" << endl;
cin >> k;
cout << " string is" << endl << k << endl;
system("Pause");
return 0; }
```

k = User ka input Name of string

output.

Ali

0	1	2	3	4
A	l	i	\0	X

\0 = Null

cin کا نقصان ہے کہ اگر space ہو تو وہ space کے بعد
والا لکھا ہوا شو نہیں کرواتا۔

cin → Ali usman. **196**

cout → Ali space سے پہلے والی چیز کو شو کروائے گا۔

② cin.get() یہ space کے بعد والا بھی
شو کروائے گا۔

```
#
int main()
{ char K[10];
cout << "Enter the string" << endl;
cin.get(K, 10);
cout << "string is" << endl << K << endl;
    system("Pause");
    return 0; }
```

both same

⇒

```
cin.get(K, 10, '*');
```

اس پر ہم (Bound) پابند ہو جاتے ہیں کہ ⑩ string ہونے تو
compiler ہمیں output دکھائے گا۔
مگر اگر ہم اس لفظ کے ساتھ وہی character لگا دے تو output آجائے گا

eg Junaid Bilal ⇒ Junaid*

output Junaid ✓

Program Intialization 107

(14)

```
# —
int main()
{ char k[20] = "Ghauricenter";
cout << "string is <<endl<< k<<endl;
      System("Pause");
      Return 0; }
```

اگر ہم [ ] کو خالی چھوڑ دے تو بھی یہ output آ جائے گی
Automatic یہ string کی length کو Adjust کرتا ہے۔

⇒ 2nd way of intialization

```
char k[20]={'G','h','a','u','r','i',' ','c','e','n','t','e','r'};
```

اس میں length لازمی دینا ہے۔

Syntax char array_name[length] = Value;

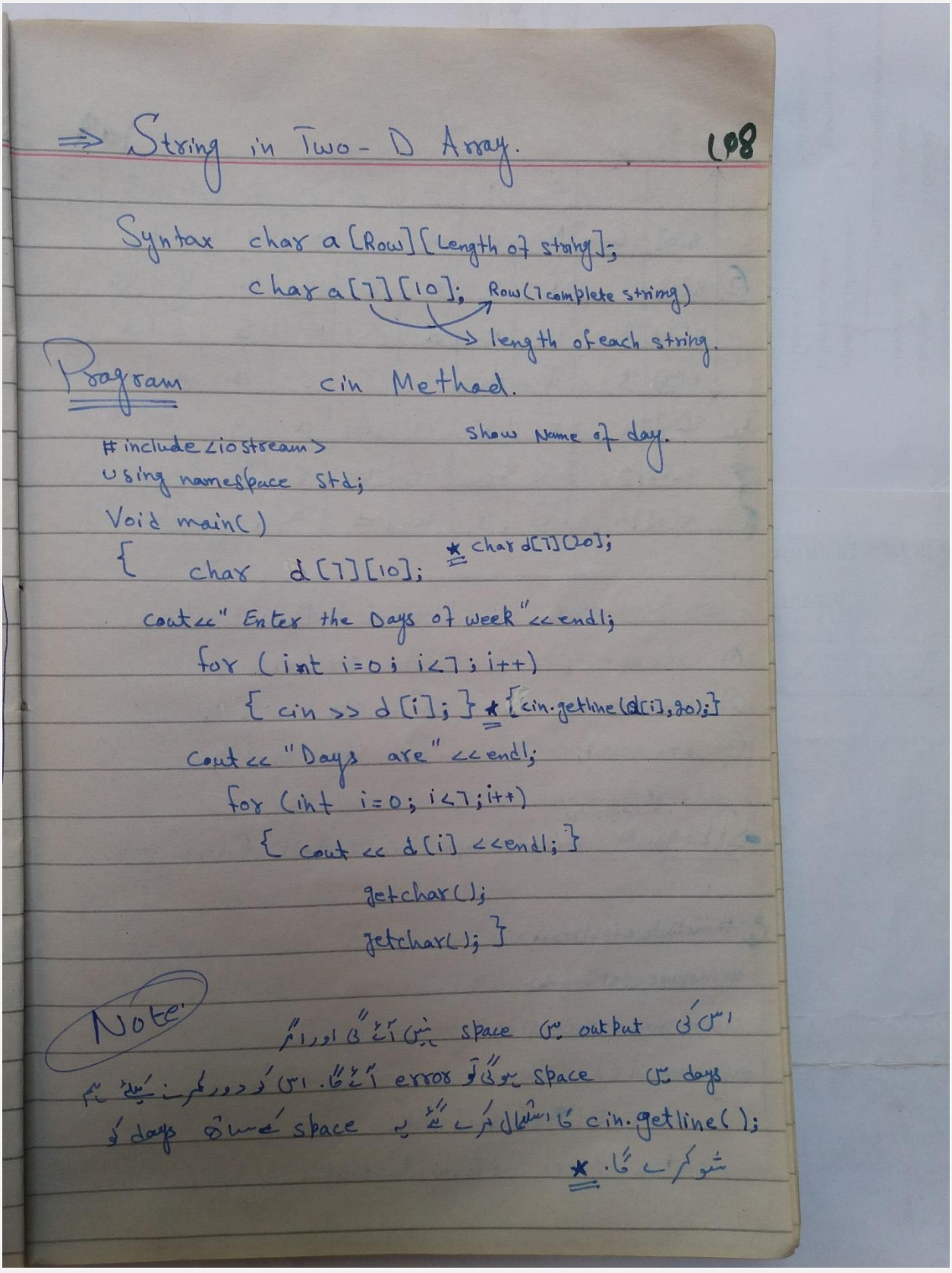
---

information.

```
int a=10;
float b=10.5;
Char z='j';
String a[]={.};
```

→ بریکٹ تک بھی لکھ سکتے ہیں اور نہیں بھی

Example



⇒ String in Two-D Array.

108

Syntax char a[Row][Length of string];

char a[7][10]; Row(7 complete string)

length of each string.

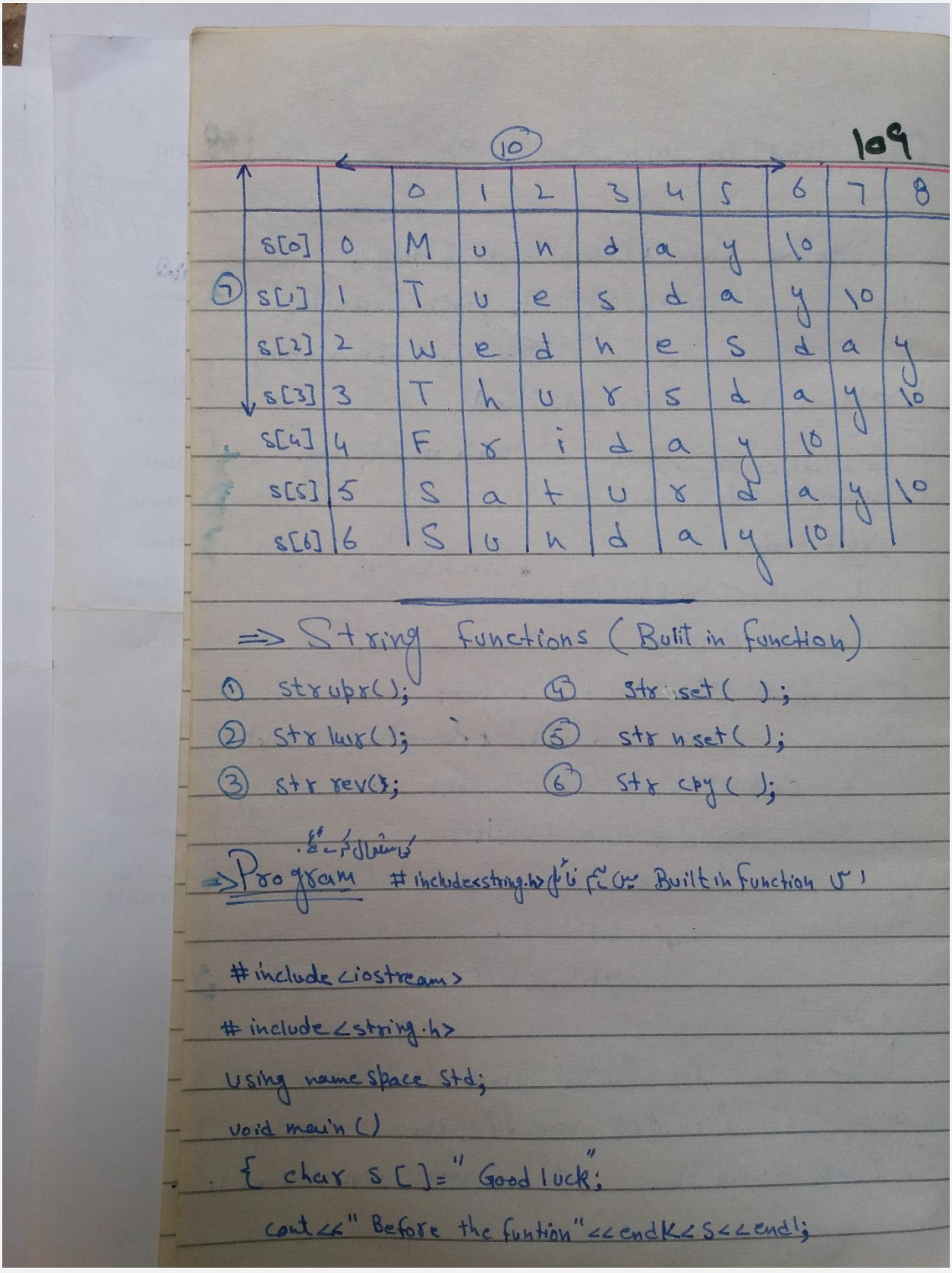
Program cin Method.

Show Name of day.

```
#include<iostream>
using namespace std;
Void main()
{   char d[7][10];   * char d[7][10];
cout<<" Enter the Days of week"<<endl;
   for (int i=0; i<7; i++)
     { cin >> d[i]; } * {cin.getline(d[i],20);}
cout<< "Days are"<<endl;
   for (int i=0; i<7;i++)
     { cout << d[i] <<endl; }
        getchar();
        getchar(); }
```

Note:

اس کی output میں space نہیں آئے گی اور اگر space دی تو error آئے گا۔ اس کو دور کرنے کے لئے days کے space میں cin.getline(); استعمال کرے گا۔ شو کرے گا۔ *



109

(10)

		0	1	2	3	4	5	6	7	8
s[0]	0	M	u	n	d	a	y	10		
s[1]	1	T	u	e	s	d	a	y	10	
s[2]	2	W	e	d	n	e	s	d	a	y
s[3]	3	T	h	u	r	s	d	a	y	10
s[4]	4	F	r	i	d	a	y	10		
s[5]	5	S	a	t	u	r	d	a	y	10
s[6]	6	S	u	n	d	a	y	10		

(1)

⇒ String functions (Bulit in function)

① strupr();
② strlwr();
③ strrev();
④ strset();
⑤ strnset();
⑥ strcpy();

⇒ Program کو استعمال کرنے کے لیے #include<string.h> فائل میں Built in Function ہوتے ہیں

```
#include<iostream>
#include<string.h>
using namespace std;
void main()
{ char s[]="Good luck";
  cout<<" Before the funtion"<<endl<<s<<endl;
```

```
9   strupr(s);
    cout<<" After the function "<<endl<<s<<endl;
        getchar();
10      getchar(); }
```

(1) یہ تمام lowercase الفاظ کو upercase letter میں convert کرے گا۔

(2) یہ تمام upercase letter کو lowercase letter میں تبدیل کرے گا۔

(3) یہ الفاظ کو ریورس کرے گا۔ مثلاً Junaid ⟶ dianuJ (strrev();)

(4) **Program**

```
void main()
char s[]="Good Luck";
char z = '*';
cout<<"Before the function"<<endl<<s<<endl;
Str.set(s,z);
cout<<"After the function"<<endl<<s<<endl;
    getchar();
    getchar(); }
```

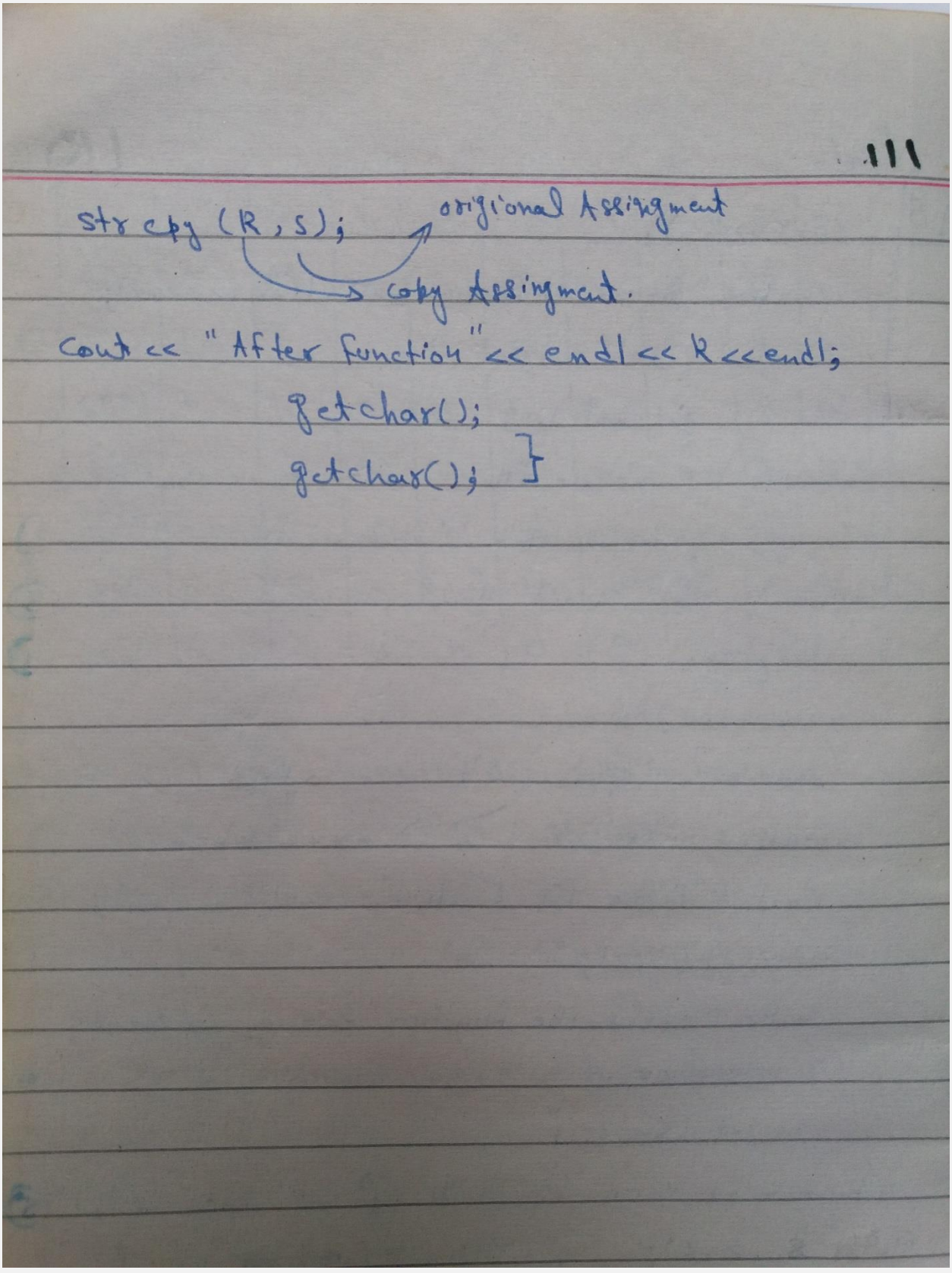
output → **** ****

(5) اس کا code بھی (4) والا رہے گا مگر اس میں 2 Parameter کی جگہ تیسرا Parameter بھی آئے گا۔

```
strnset(s , z , 2);
```

(6) output **od Luck

```
#include<iostream>
#include<string.h>
using name space std;
Void main()
{ char s[] = "Good luck";
  Char k[20];
```


111
strcpy (R, S);
origional Assignment
copy Assignment.
Cout << "After function" << endl << R << endl;
getchar();
getchar(); }

⇒ Pointers:-

112

Pointer is a variable that is used to store the memory address of other variable.

Syntax:-

Datatype space *Name of variable;

Asterisk.

int (*z);

Pointer variable

Program:- Point declaration.

```
#____
void main()
{ int *z;
  int m;
  int n;
  z = &m;
  cout << "Address of m" << endl << z << endl;
  z = &n;
  cout << " Address of n" << endl << z << endl;
  getchar();
  getchar();
}
```

Datatype values

113

⇒ Indirectional operater / Dereffference operater.

Symbol:. *

⇒ Program

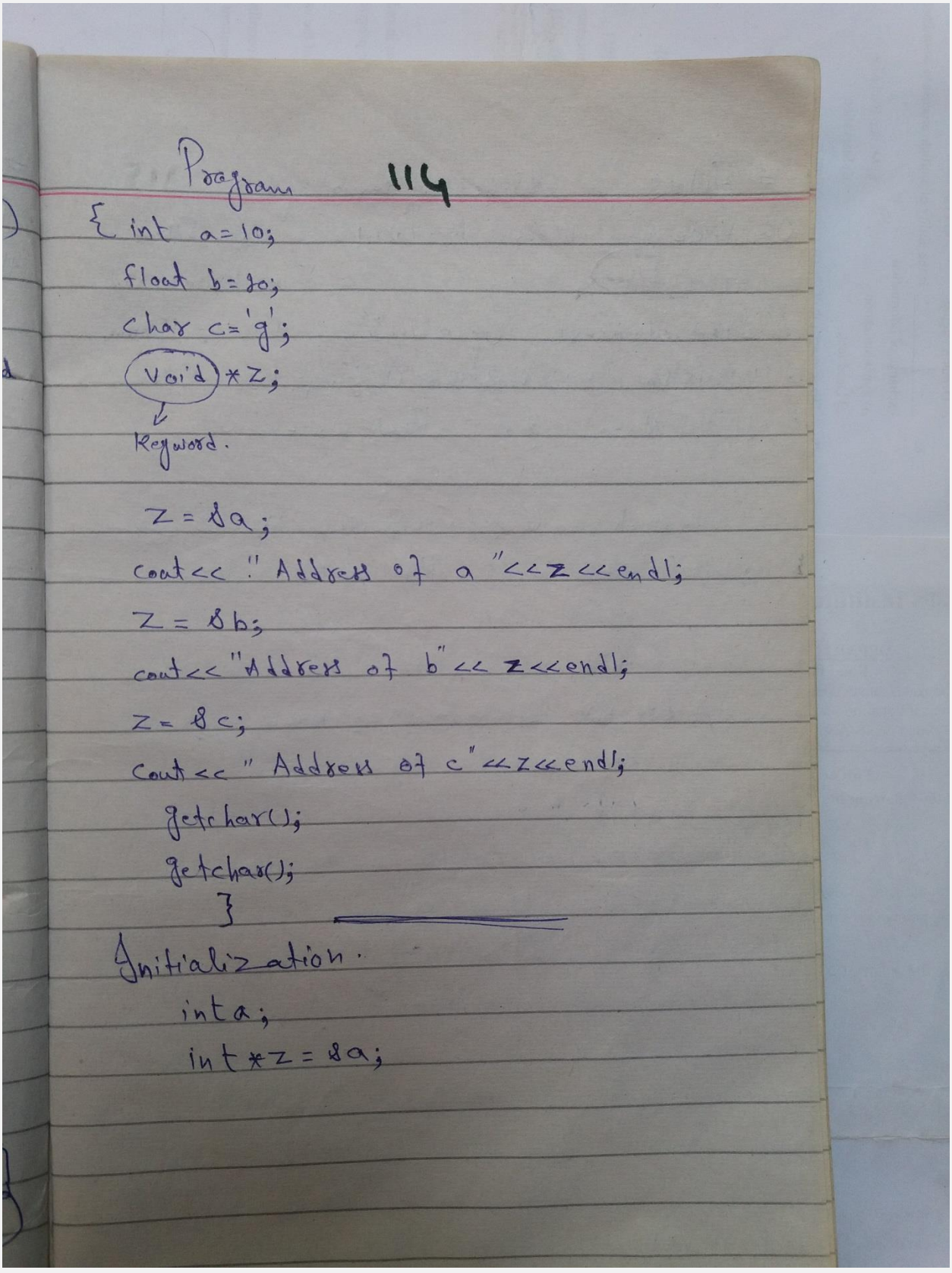
It is use to access the value of the variable whose memory Address stored in Pointer variable.

```
#=
{ int *z;
  int m=10;
  int n=20;
  z = &m
cout << "Address of m" <<endl<< z <<endl;
cout << " Value of m" <<endl<<*z << endl;
 z = &n;
cout << "Address of n" <<endl<<z<< endl;
cout << " value of n" << endl<<*z<<endl;
  getchar();
  getchar();
  }
```

⇒ Void Pointer

Program

```
#=
'void main( )
```



Program 114

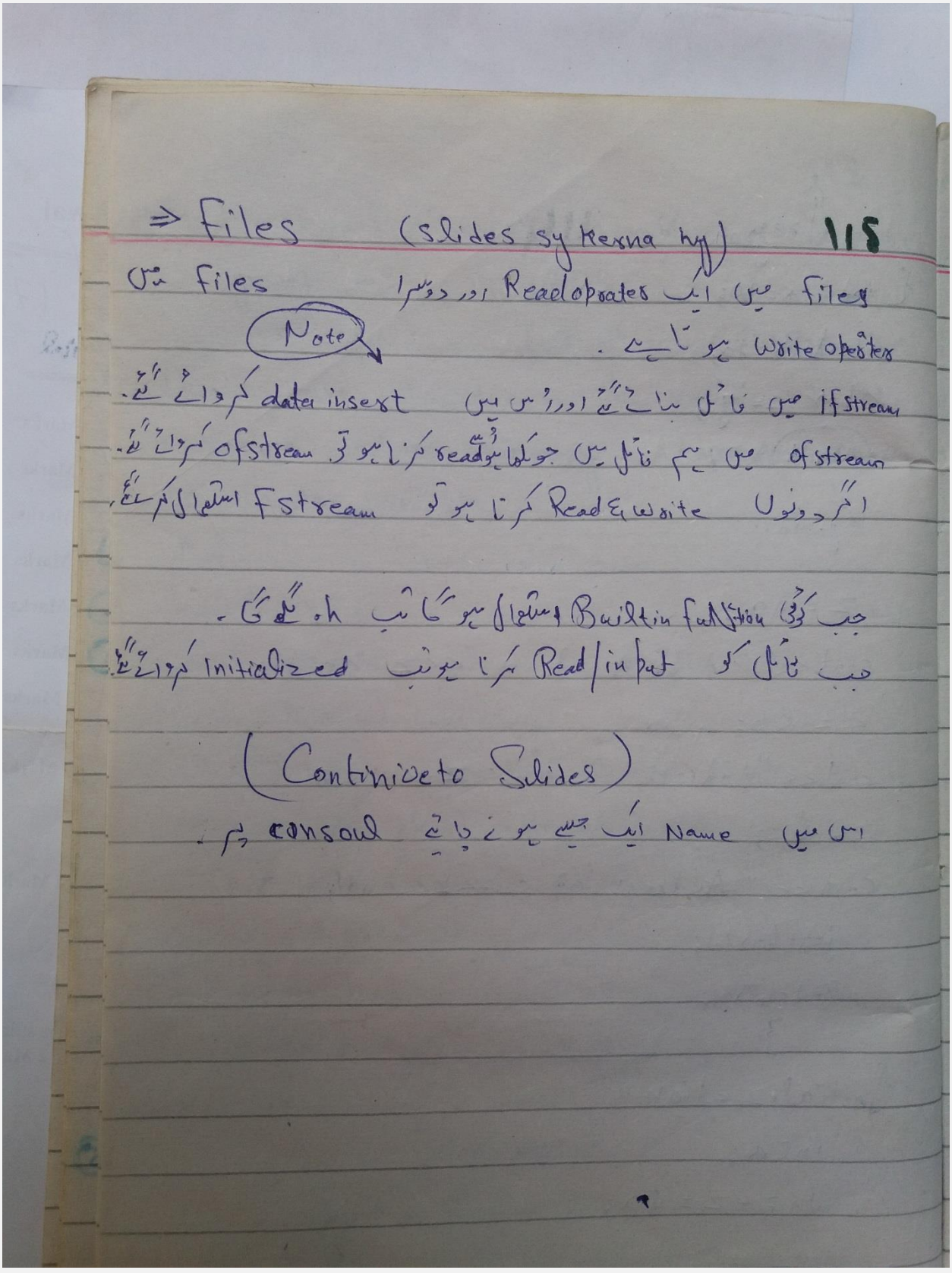
```
{ int a=10;
  float b=20;
  char c='g';
  void *z;
```

Keyword.

```
  z = &a;
  cout<< " Address of a "<<z<<endl;
  z = &b;
  cout<< "Address of b"<< z<<endl;
  z = &c;
  cout<< " Address of c"<<z<<endl;
    getchar();
    getchar();
      }
```

Initialization.

```
  int a;
  int *z = &a;
```



⇒ Files (slides sy krna hy) 115

Files میں ایک Read operator اور دوسرا Files میں
Write operator ہوتا ہے۔

Note

data insert کروانے کے لئے ifstream میں فائل بنائے جاتے اور اس میں
ofstream میں ہم فائل میں جو کچھ ہو read کرنا ہو تو ofstream کروائے جاتے۔
اگر دونوں Read & Write کرنا ہو تو Fstream استعمال کرتے۔

جب کوئی Builtin function استعمال ہوتا تب h. لگتی۔
جب فائل کو Read/input کرنا ہو تب Initialized کروائے جاتے۔

(Continue to Slides)

اس میں Name ایک جیسے ہونے چاہئے consoul پر۔

"Structures" 116

Structures is the Combination | Collection of different datatype with same name.

(i) Declaration (ii) Defining (iii) Accessing.

(i) Declaration the Sturctures

Syntax.

```
Struct  space  Name of structure
{ Datatype1  space  variable1;
  Datatype2  space  variable2;
      :              :
  Datatypen  space  Variablen};
```

Example

```
Struct  Student
{int    roll no;
 int    mark;
};
```

Variable کی جگہ structures میں جو

→ Datmember (or) Member Variable

(2) Defining the structures:

Syntax:

Name of structure Structure Variable;

Example Student S;

117

Calling Funtion But in Structures we say Accessing Data member/ Member variable

Syntax:

Structure variable . Member variable;

Example S. rollno;

S. mark;

اصل میں Declaration اور defining the structures کے Accessing

The structures کے ساتھ ملا کر Combine کیا جاتا ہے۔

Program Declaration

Strrecter میں ہم سب سے ہمیشہ
Before the main function کو لکھا کرتے
ہیں۔

```
# include <iostream>
using namespace std;

struct car          --> Declartion
{ int model no;
  int Part no;
  float cost;
};

void main()
{ car c1, c2;       --> funtion defining
cout << "Enter the model no" << endl;
cin >> c1.modelno   --> Accessing
```

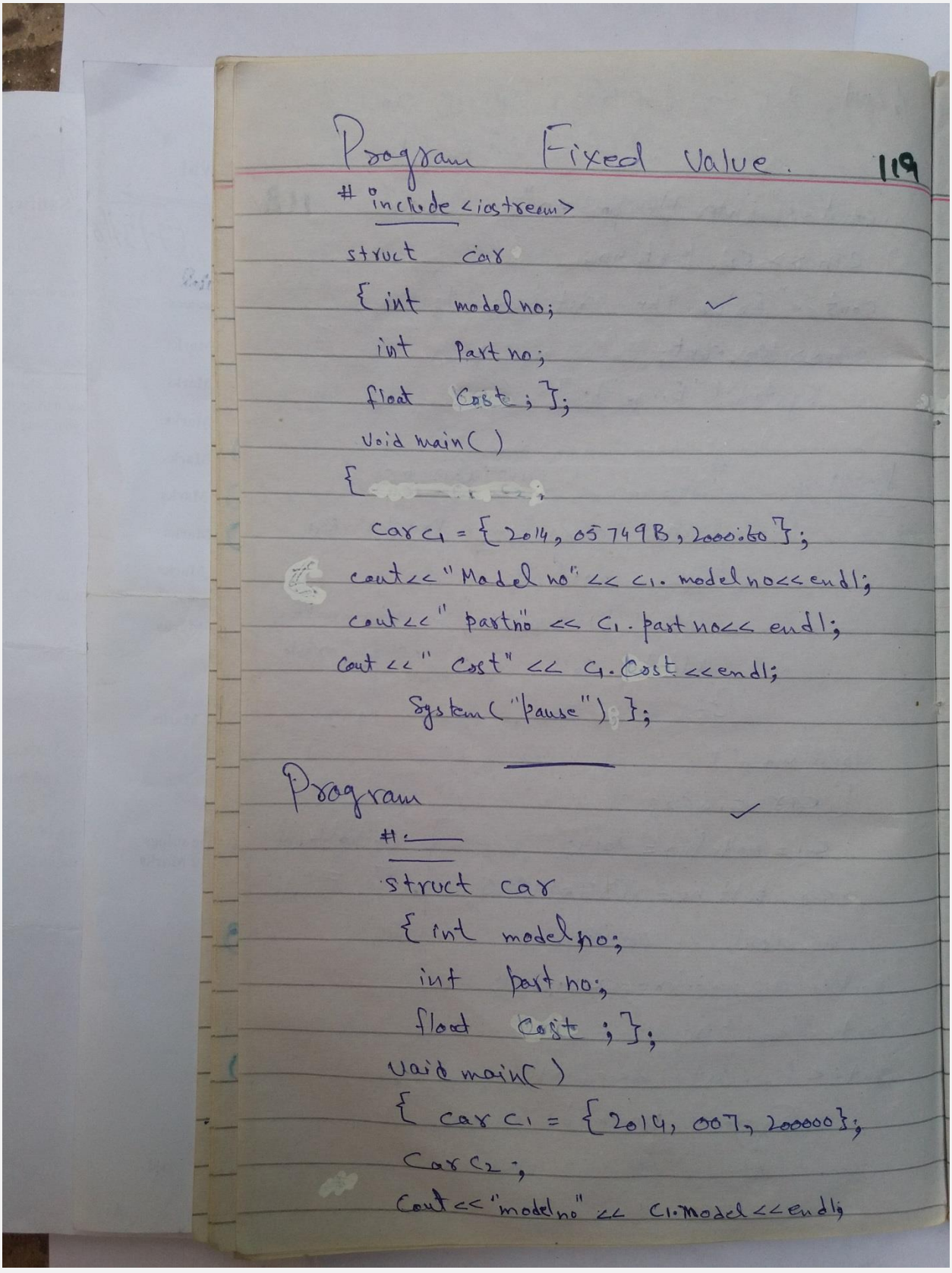
اگر cin میں c1 کو استعمال کرتے ہیں تو cout میں
بھی c1 ہی لکھنا ہے۔


```
cout<<"Enter the part no"<<endl;
 cin>>c2.part no;
cout<<"Enter the cost"<<endl;
cin>>c2.cost;
    System("Pause"); }
```

Program # ___ using ___ (initialized.)

```
struct car            --> Declaration.
{int   modelno;   ]
 int   part no;   ]  --> Data member/
 float Cost;      ]      member variable.
};

Void main()
 { car c1, c2;
   c1.modelno = 2014;          ]
   c2.part no = J89017B5;      ]  --> initialized
   c2.Cost = 200000;           ]
cout<<"c1.modelno"<<endl;
Cout<<"c2.part no"<<endl;
Cout<<"c2.Cost"<<endl;
      System("pause"); }
```

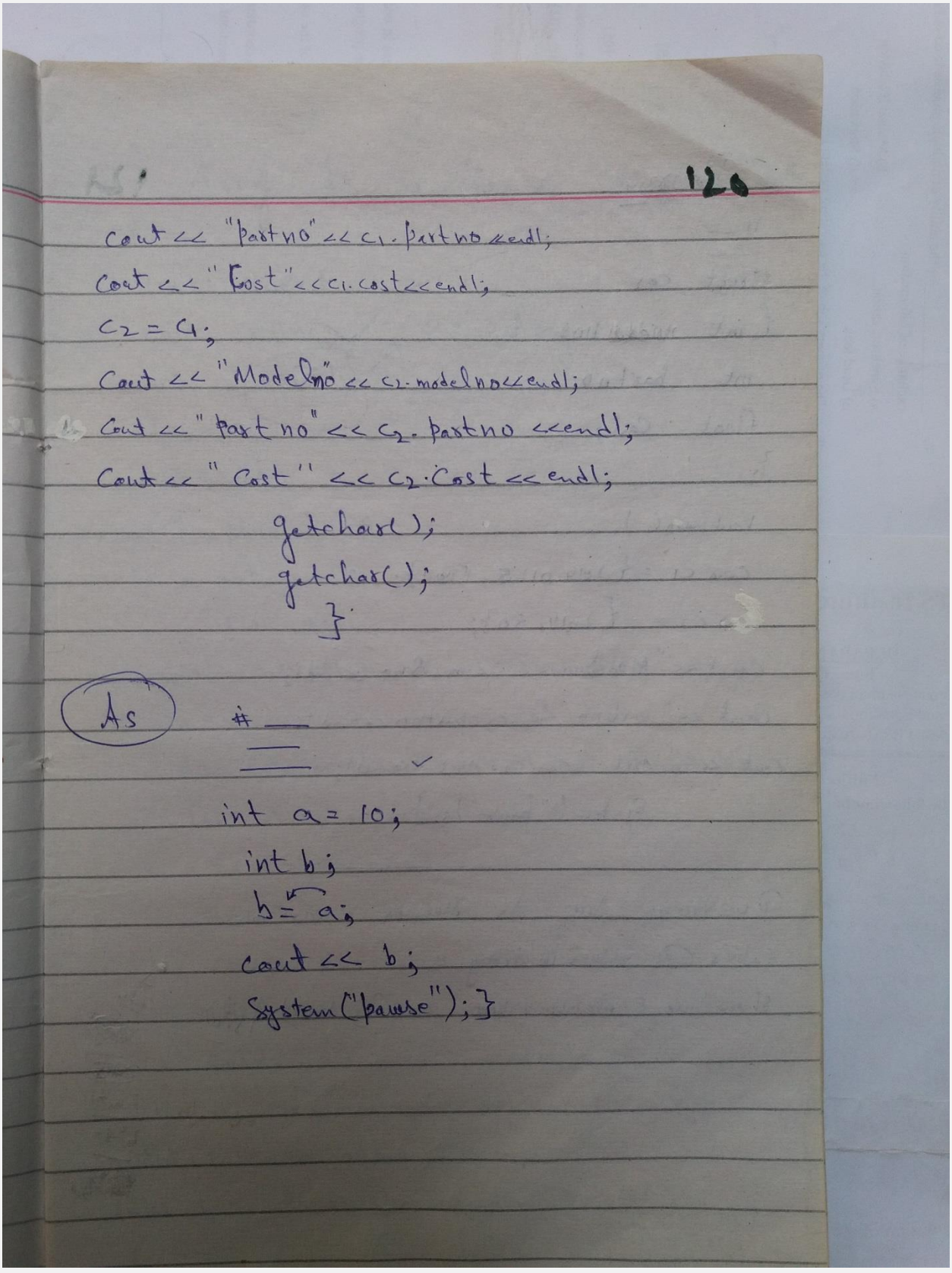


Program Fixed value. 119

```
#include <iostream>
struct car
{int modelno;
 int partno;
 float cost;};
void main()
{
 car c1 = {2014, 05749B, 2000.60};
cout<<"Model no" << c1.modelno<<endl;
cout<<" partno" << c1.partno<< endl;
cout<<" cost" << c1.cost<<endl;
 system("pause");};
```

Program

```
#
struct car
 {int modelno;
  int partno;
  float cost;};
void main()
 { car c1 = {2014, 007, 200000};
 Car c2;
 Cout<<"modelno" << c1.model<<endl;
```



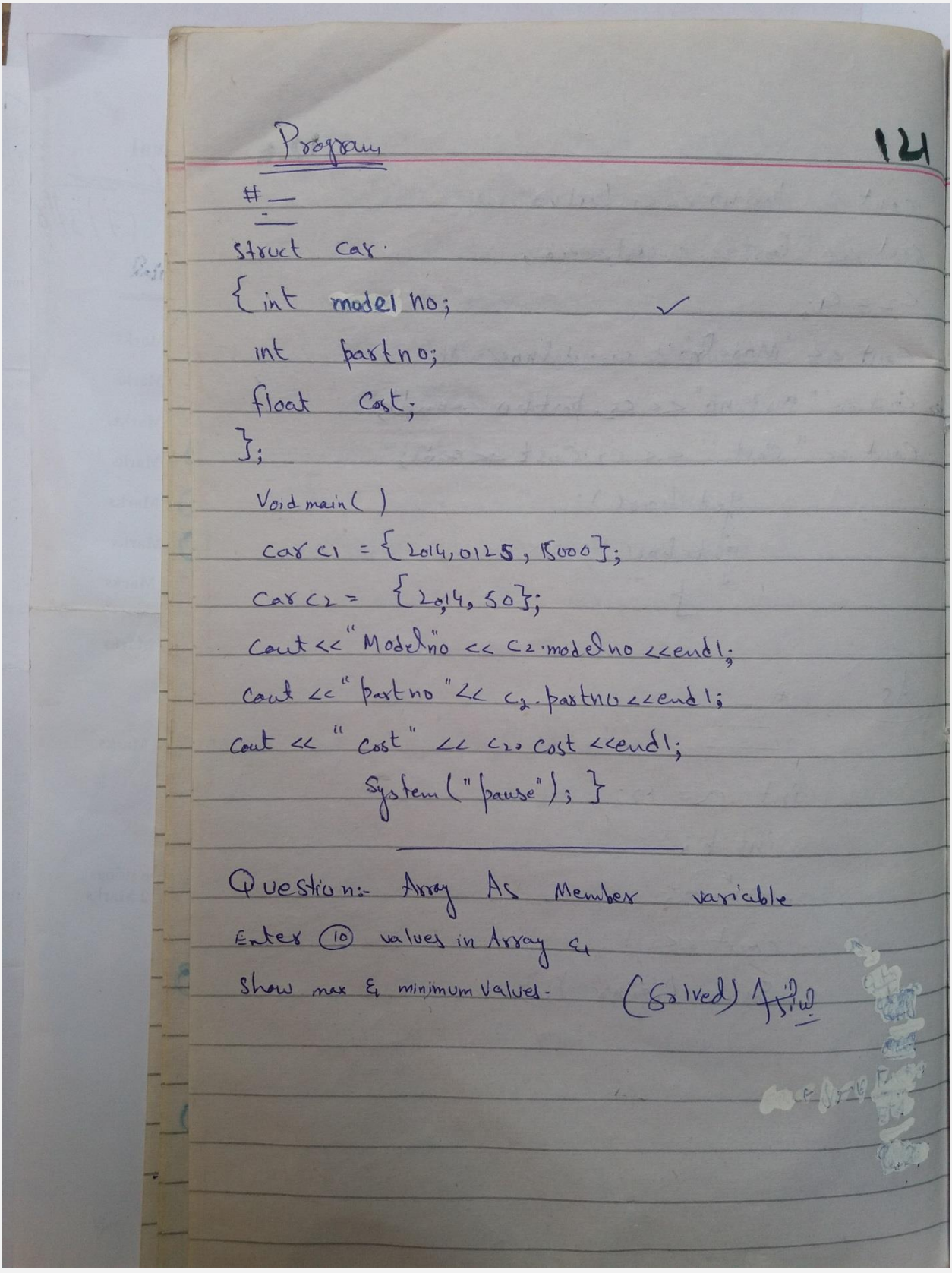
120

```
cout << "partno" << c1.partno << endl;
cout << "Cost" << c1.cost << endl;
c2 = c1;
cout << "Modelno" << c2.modelno << endl;
cout << "partno" << c2.partno << endl;
cout << "Cost" << c2.Cost << endl;
        getchar();
        getchar();
        }
```

As

#

```
int a = 10;
int b;
b = a;
cout << b;
system("pause"); }
```



Program 14

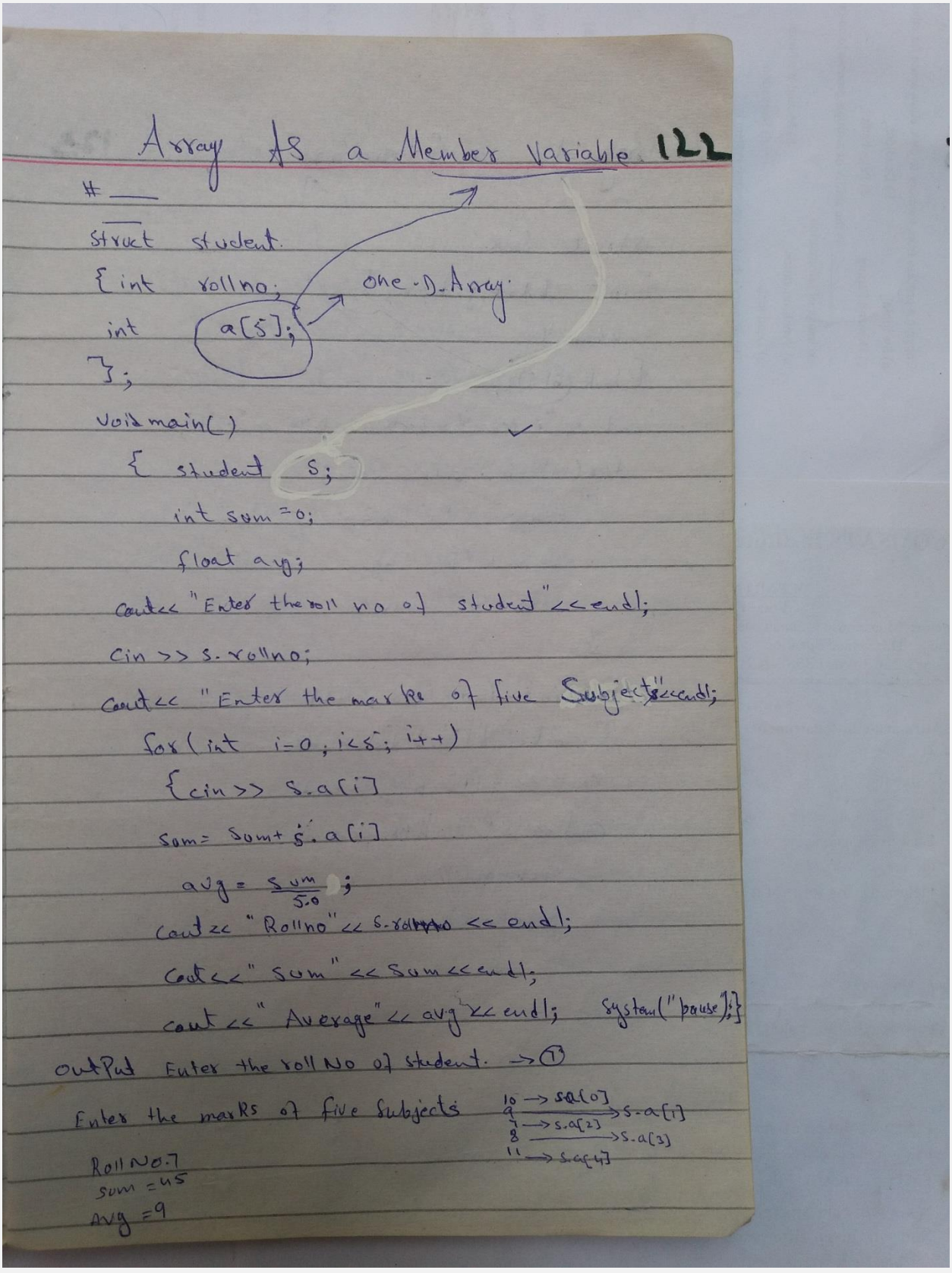
#

```
struct car
{ int model no;
  int partno;
  float cost;
};
Void main( )
 car c1 = {2014, 0125, 5000};
 car c2 = {2,14, 50};
Cout << "Model no << c2.model no <<endl;
Cout << "partno " << c2.partno <<endl;
Cout << " cost " << c2.cost <<endl;
        System ("pause"); }
```

Question:- Array As Member variable

Enter (10) values in Array &

Show max & minimum values. (Solved) Trw



Array As a Member Variable 122

```
#___
struct student
{ int rollno;
  int a[5];
};
void main()
  { student s;
    int sum=0;
    float avg;
cout<< "Enter the roll no of student"<<endl;
cin >> s.rollno;
cout<< "Enter the marks of five Subjects"<<endl;
  for(int i=0; i<5; i++)
   {cin >> s.a[i]
   sum= sum+ s.a[i]
    avg= sum/5.0;
cout<< "Rollno"<< s.rollno << endl;
cout<<" sum" << sum<<endl;
cout<<" Average"<< avg<<endl;  system("pause");}
```

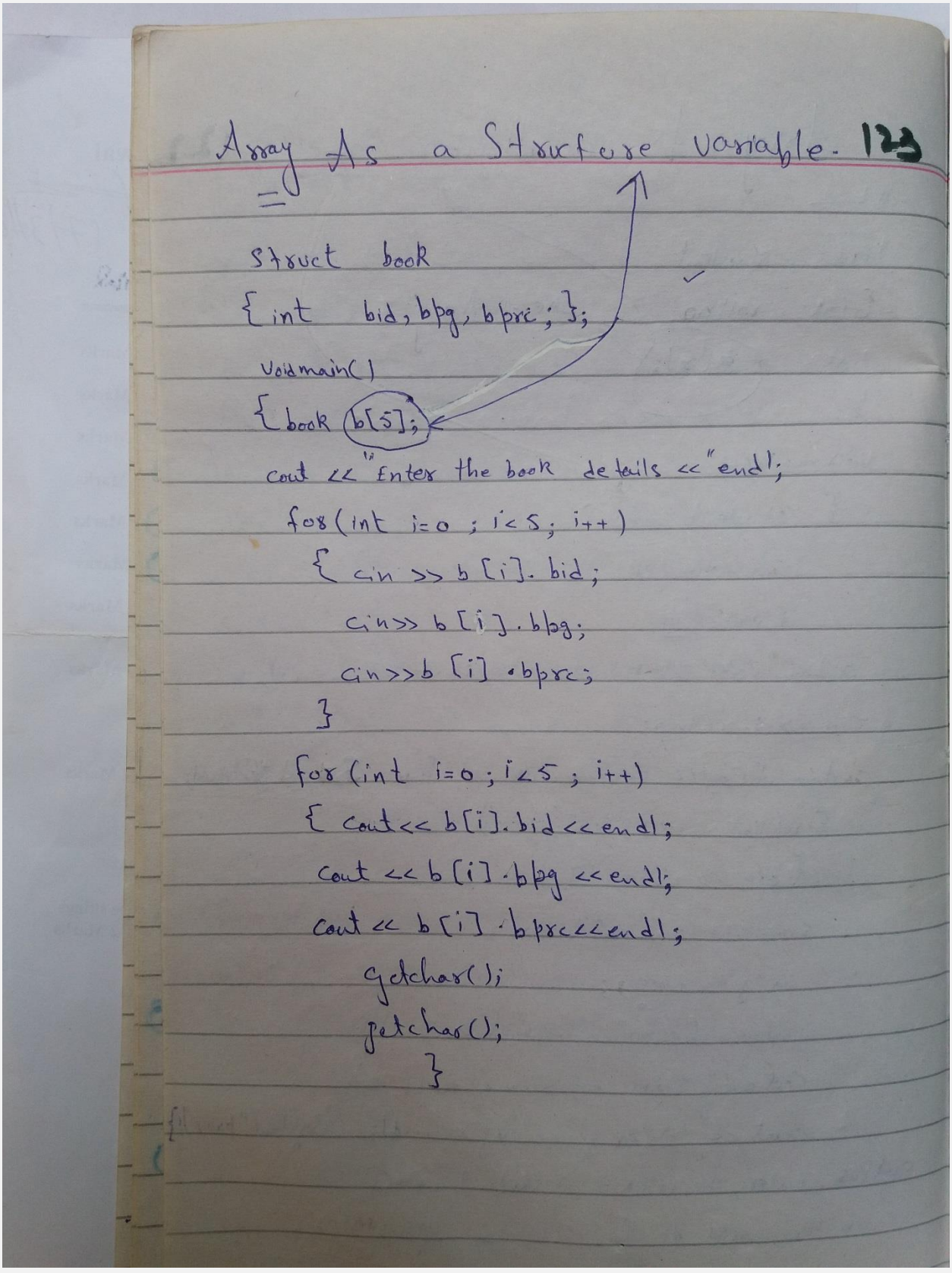
one-D-Array

OutPut Enter the roll No of student. → (1)

Enter the marks of five subjects

10 → s.a[0]
9 → s.a[1]
7 → s.a[2]
8 → s.a[3]
11 → s.a[4]

Roll NO.7
sum = 45
Avg = 9



Array As a Structure variable. 123

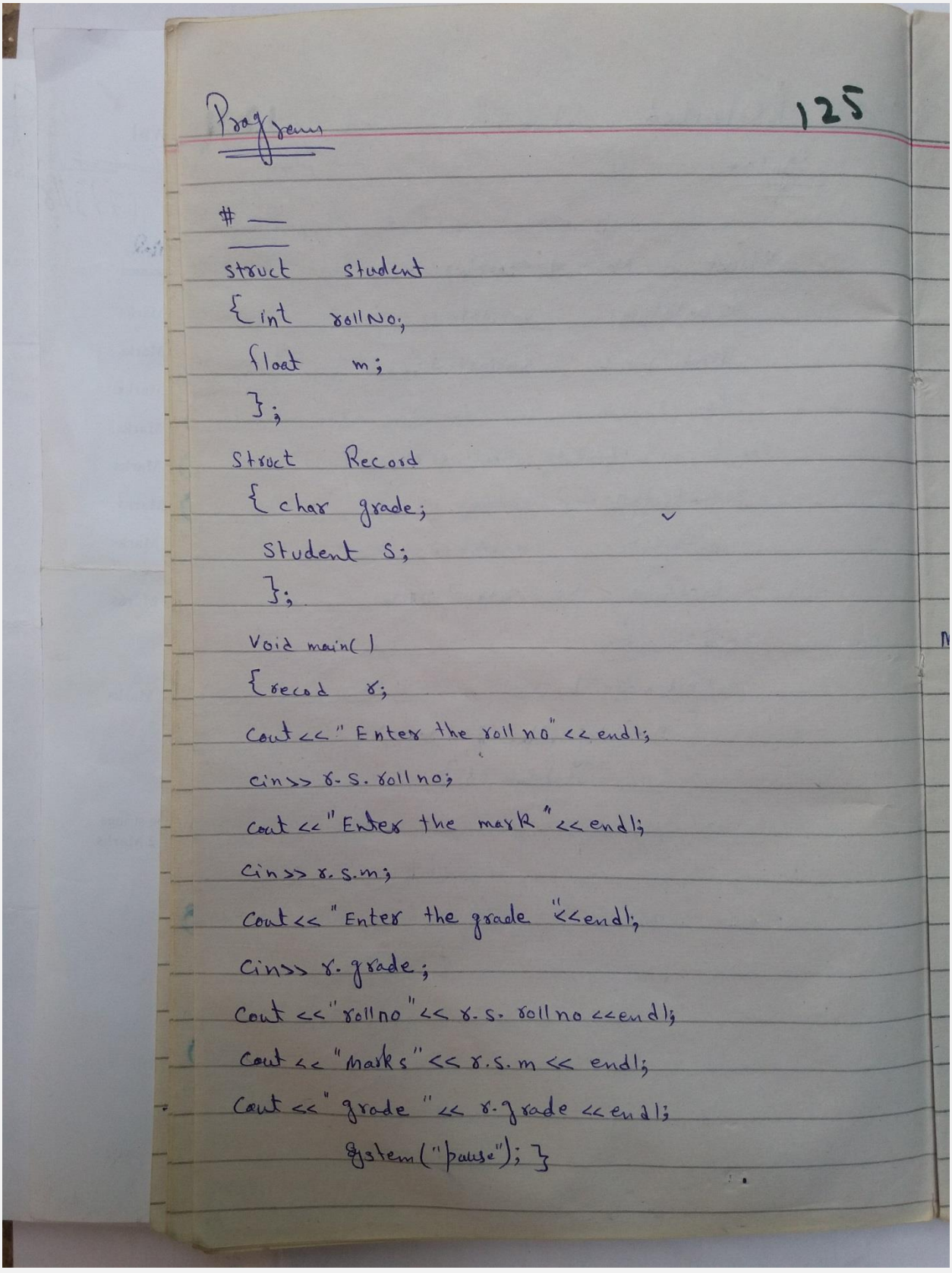
```
struct book
{ int bid, bpg, bprc; };
void main()
{ book b[5];
cout << "Enter the book details" << "end";
for(int i=0 ; i<5; i++)
  { cin >> b[i].bid;
    cin>> b[i].bpg;
    cin>>b[i].bprc;
  }
for (int i=0 ; i<5 ; i++)
{ cout<< b[i].bid << endl;
cout << b[i].bpg << endl;
cout << b[i].bprc<<endl;
   getchar();
   getchar();
     }
```

Nested Structures.

124

Syntax

```
Struct    Name of structure 1
 {Datatype1    variable1;
  Dat type2    variable2;
 };
Struct    Name of structure 2
 {Datatype1    variable1;
  Data type 2    variable2;
  Defining the structure1;
 };
  Void main()
  {Defining the structure 2;
     =Statements }
```



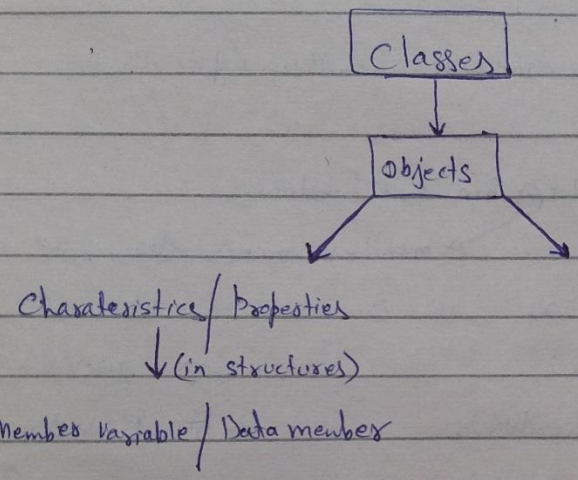
Programs 125

#—

```
struct student
{int rollNo;
 float m;
};
Struct Record
{char grade;
 Student S;
};
Void main()
{record r;
Cout<<" Enter the roll no"<<endl;
cin>>r.S.rollno;
Cout<<"Enter the mark"<<endl;
Cin>>r.S.m;
Cout<<"Enter the grade "<<endl;
Cin>>r.grade;
Cout<<"rollno"<<r.S.rollno<<endl;
Cout<<"marks"<<r.S.m<<endl;
Cout<<" grade "<<r.grade<<endl;
        System("pause"); }
```

Classes:- 126

Collection of objects with same properties & functions. class is user data type.

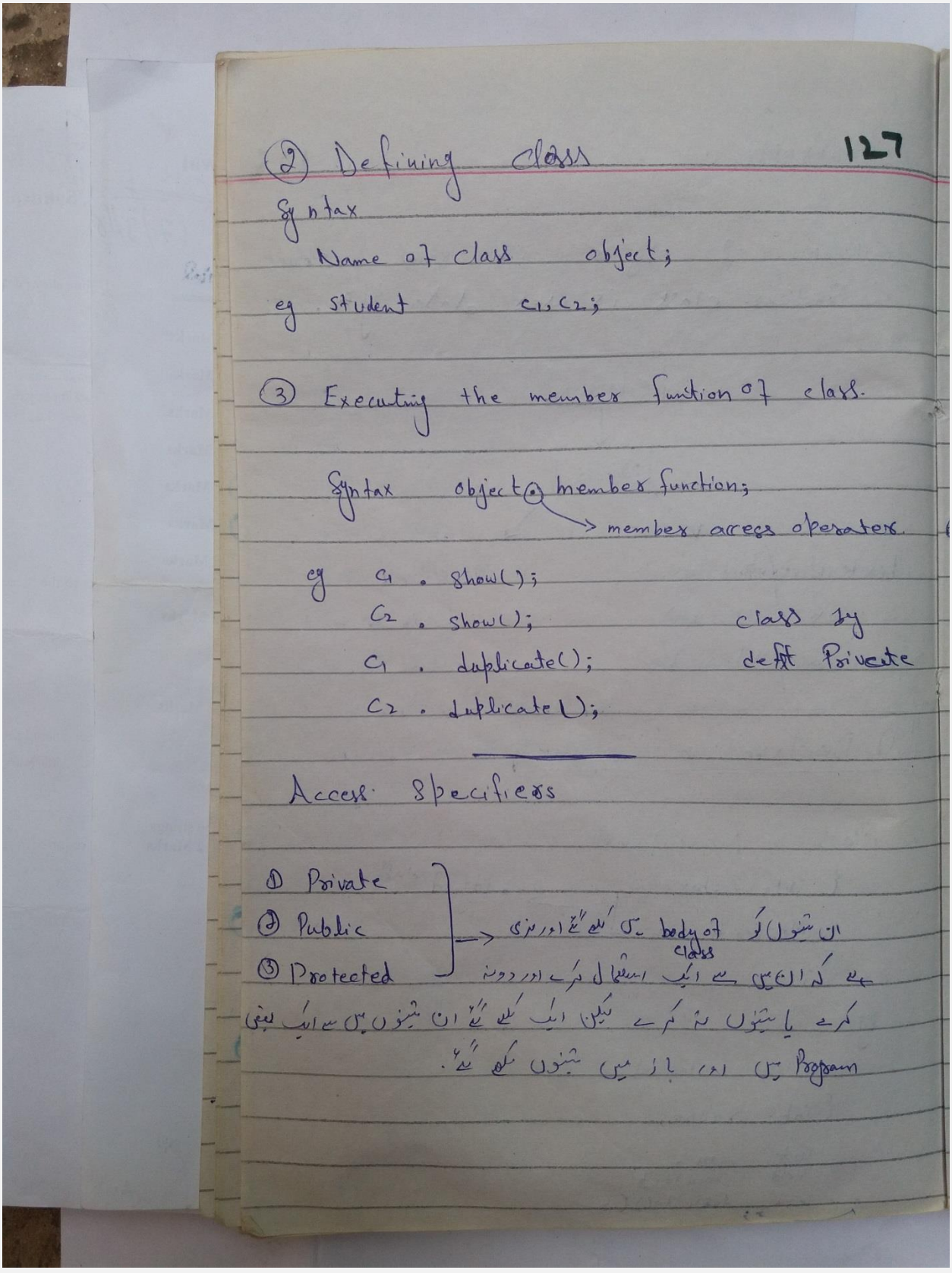


(1) Declaration. Syntax

```
class    Name of class
  {  Data member
     Member function   } ---> body of class.
  };
```

eg

```
class   student.
{ int   rollNo;
  int   m;
  void  show(){}
  void  dublicate(){}
};
```



127

② Defining class

Syntax

Name of class object;

eg Student c1, c2;

③ Executing the member function of class.

Syntax object . member function;

→ member access operater.

eg c1 . show();
c2 . show();
c1 . duplicate();
c2 . duplicate();

class by deft Private

Access Specifiers

① Private
② Public
③ Protected

→ ان شبوں کو body of class میں لکھے گئے اور بتائی ہے کہ ان میں سے ایک استعمال کرے اور دوسرے کرے یا شبوں نہ کرے لیکن ایک لکھے گئے ان شبوں میں سے ایک بتائی Program میں اور باقی میں شبوں لکھے گئے۔

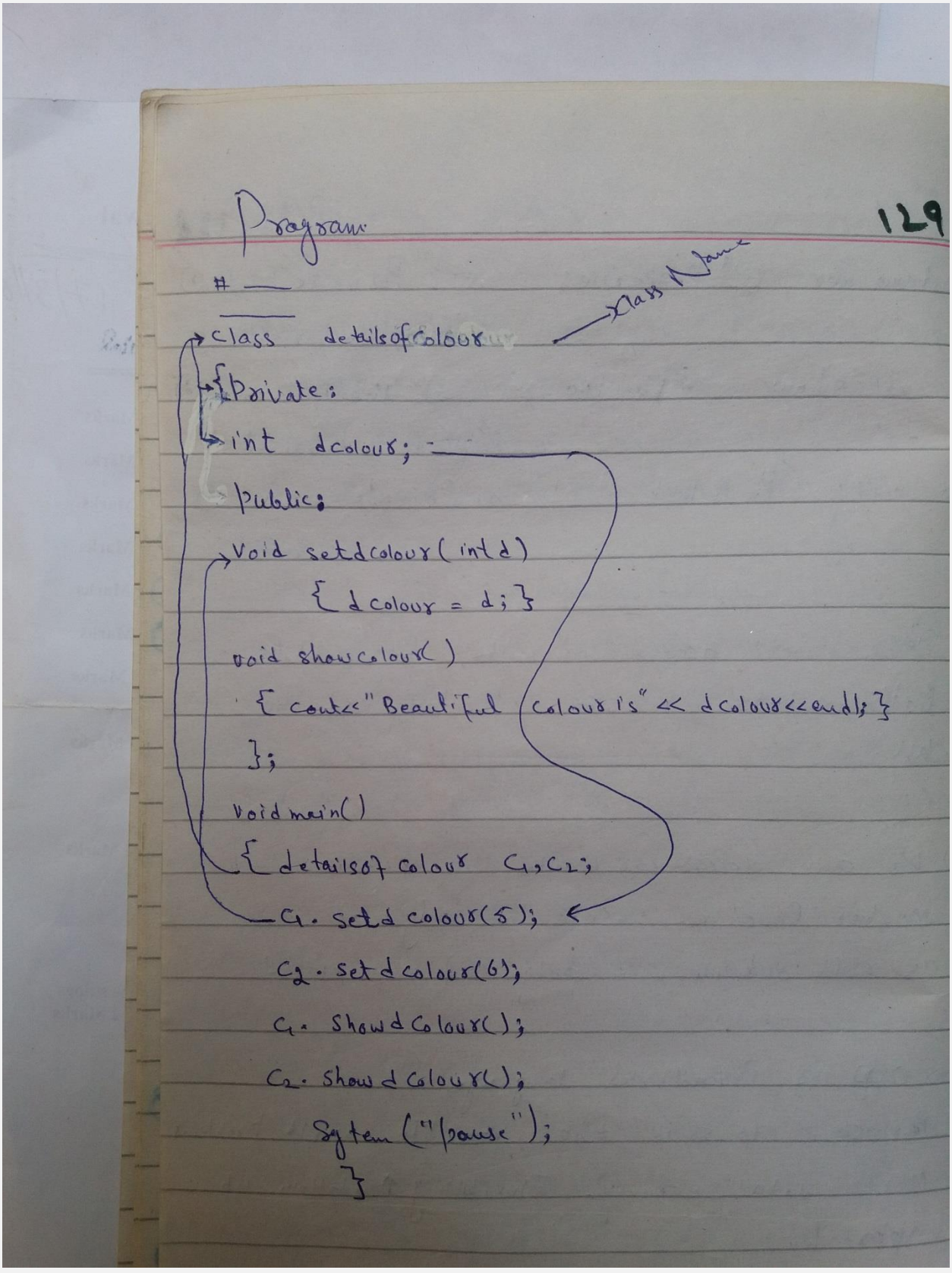
Rule 128

Datamember کو properties کہتے ہیں Private (1)
Member variable کو بھی کہتے ہیں
show کو function کہتے ہیں Public (2)
Functionality کو Protected duplicate کہتے ہیں
Private (3)
Same

(1) We cannot access or write the private Data members out side the body of the class.

(2) We can access or write the public Member function inside the body of class as well out side the body of the class.

=> OOP is Procedural language.
Perpose is to solve the real life world problems
More data security. Flexibility ↑, bottom up approch.

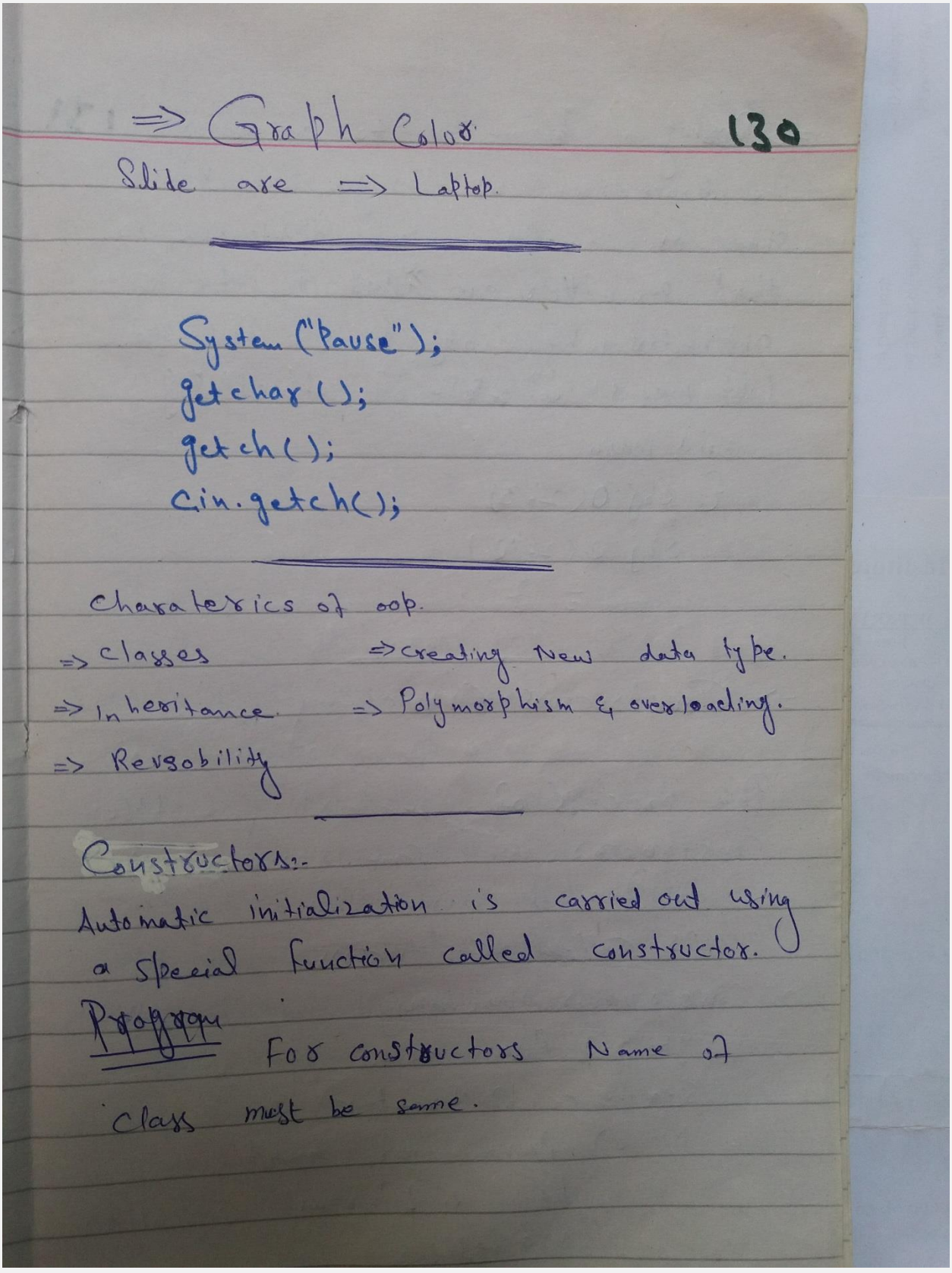


Program 129

#

Class Name

```
class detailsofcolour
{private:
int dcolour;
public:
void setdcolour(int d)
    { dcolour = d; }
void showcolour()
 { cout<<"Beautiful colour is"<< dcolour<<endl; }
};
void main()
{ detailsof colour C1,C2;
  C1.setd colour(5);
  C2.setd colour(6);
  C1.Showd Colour();
  C2.Showd colour();
    System("Pause");
    }
```



⇒ Graph Color 130

Slide are ⇒ Laptop.

System ("Pause");
getchar ();
getch ();
cin.getch();

Charaterics of oop.

⇒ classes ⇒ creating New data type.

⇒ Inheritance ⇒ Polymorphism & overloading.

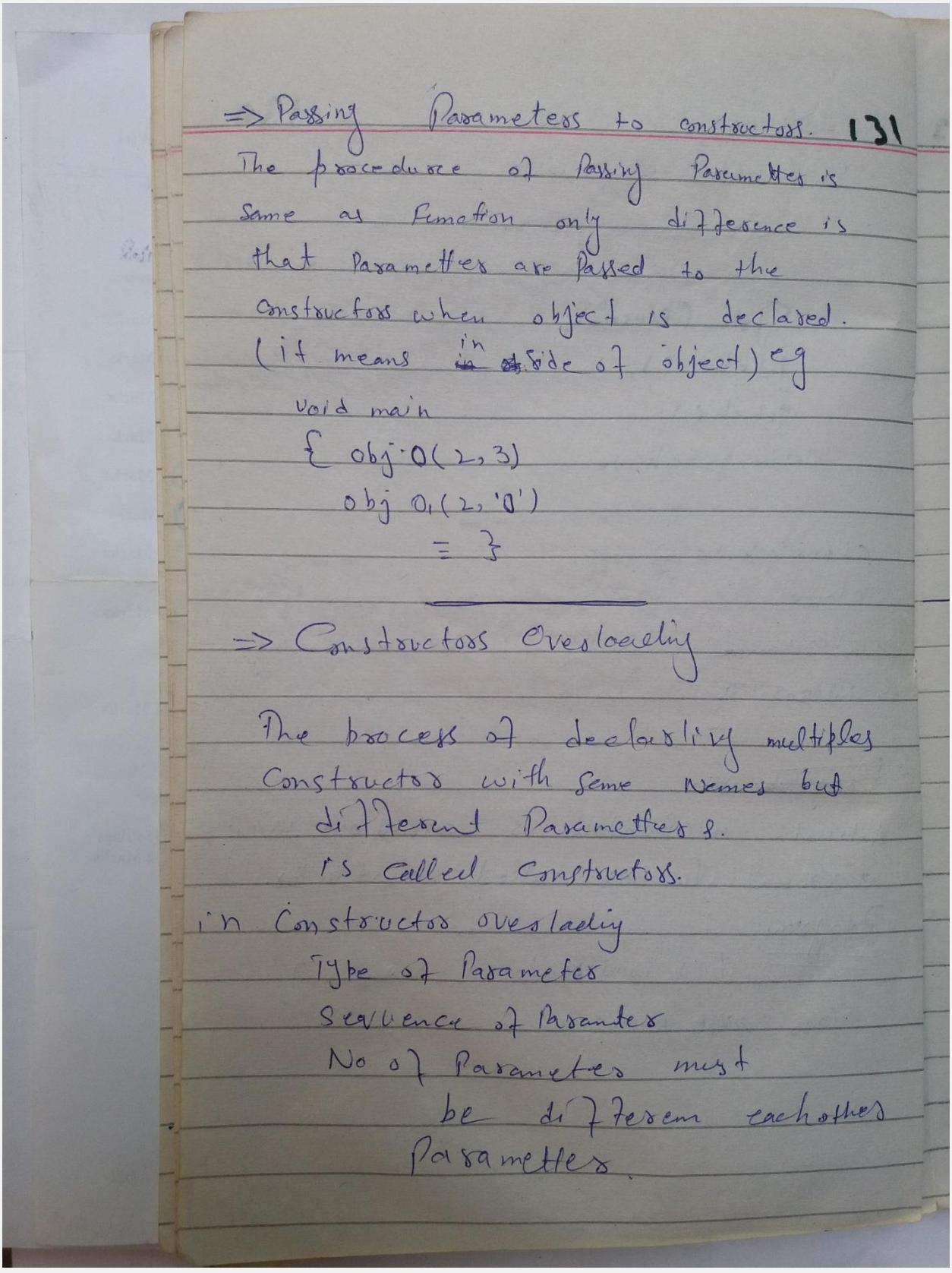
⇒ Revsobility

Constructors:-

Automatic initialization is carried out using a special function called constructor.

Program

For constructors Name of class must be same.



=> Passing Parameters to constructors. 131

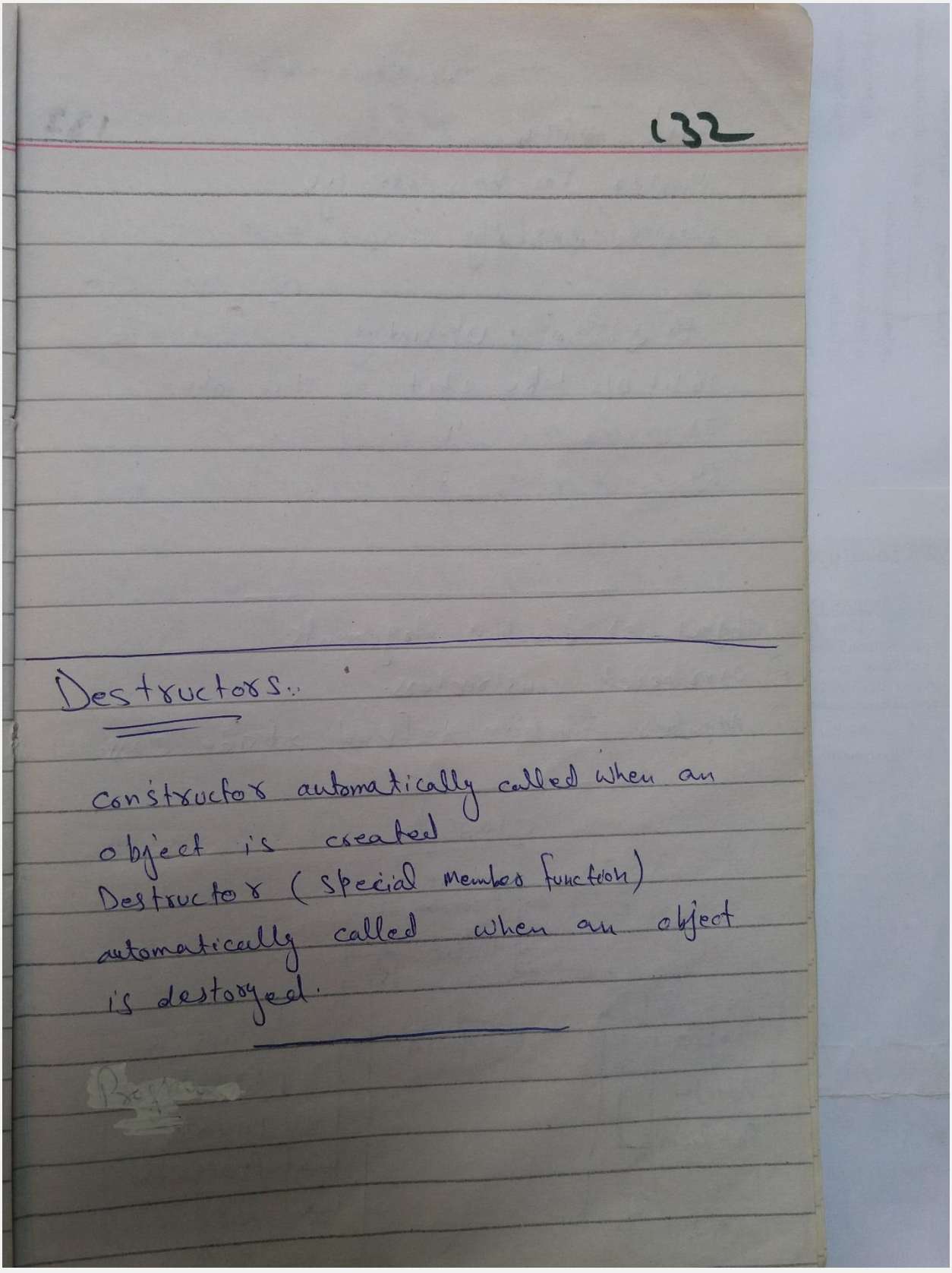
The procedure of passing Parameters is same as Fumction only difference is that Parameters are passed to the constructors when object is declared. (it means in side of object) eg

```
void main
{ obj.O(2,3)
  obj O1(2,'O')
  = }
```

=> Constructors Overloading

The process of declaring multiples Constructor with same Names but different Parameters is called Constructors.

in Constructor overloading
Type of Parameter
Sequence of Parameter
No of Parameters must be different eachothers Parameters.


132
Destructors:
Constructor automatically called when an object is created
Destructor (special member function) automatically called when an object is destoryed.

This asterik means Pointer to

int *Ptu

133

Pointer has no data type

اس بات پر منحصر ہے کہ آپ کتنی capacity کی value کو store کروا رہے ہیں

variable ، Operating پر منحصر کرتا ہے

16 bit O/S take 2bit. to store value

32 // // //. 4 // // //

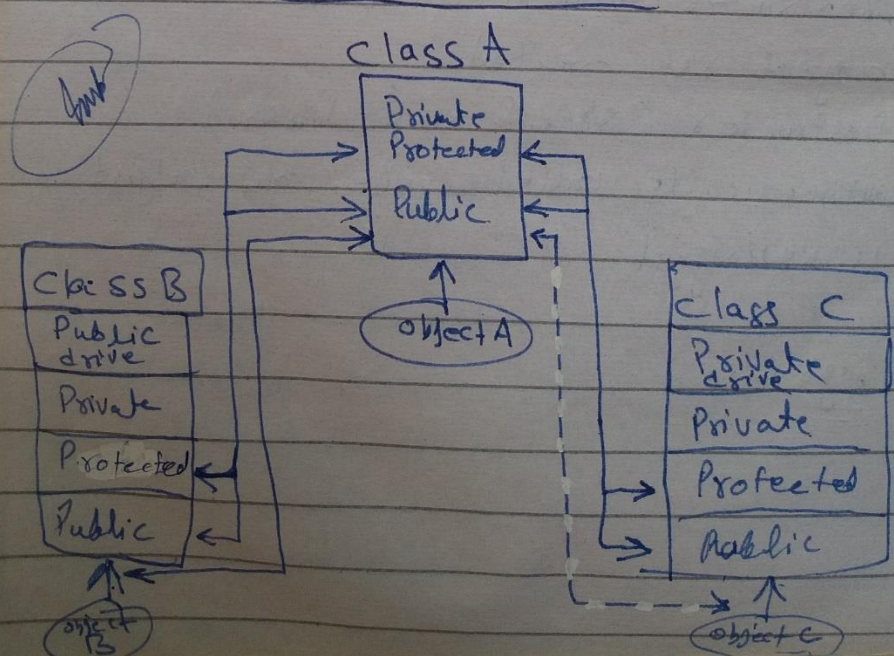
64 // // // // 8 // // // //

---

① Object as Function Arguments.

② Overloaded constructors.

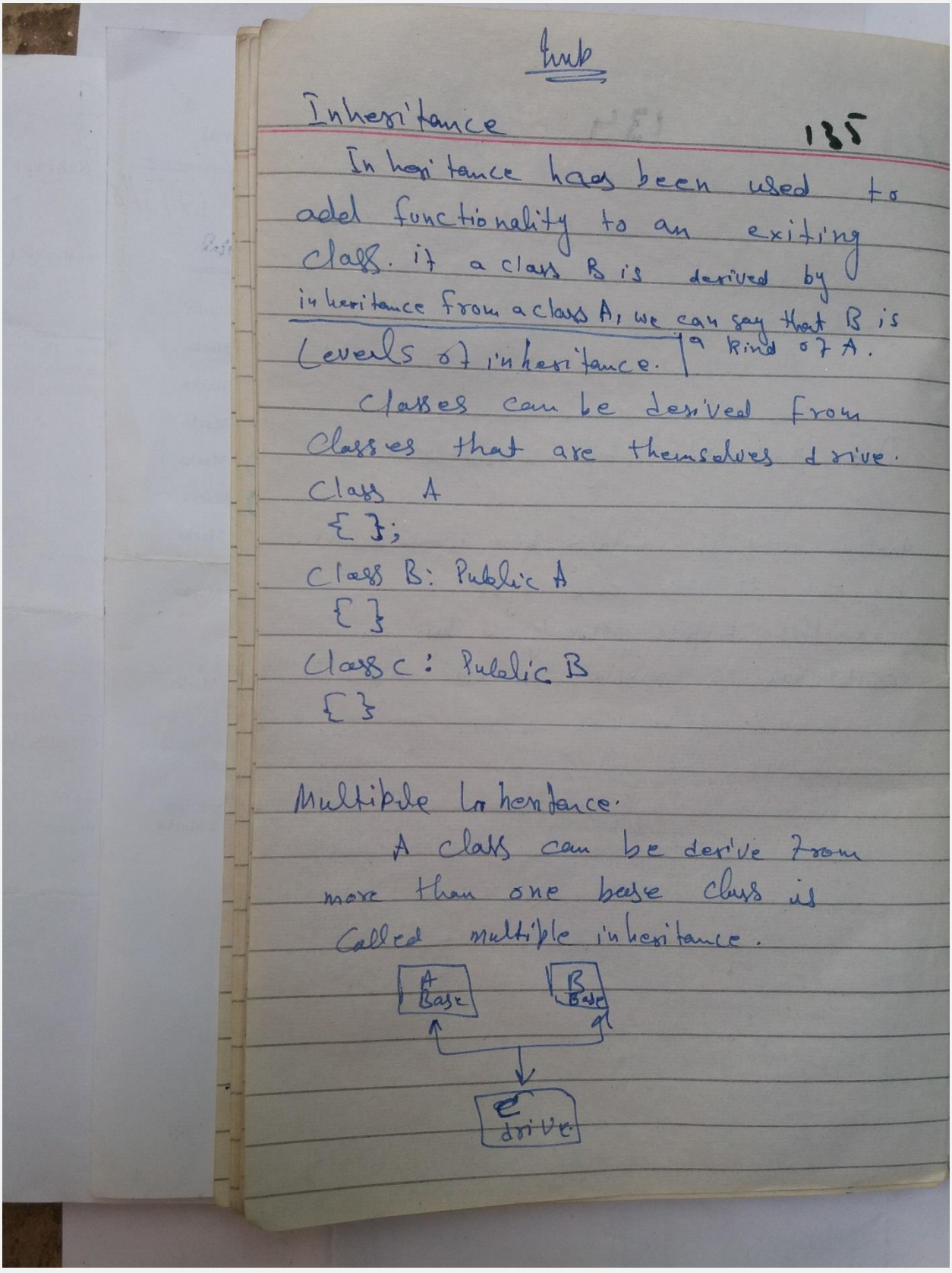
③ Member Function defined outside class.



134

The Default copy constructor.

Another way to initialization with another object of the same type.
For this purpose Already a buit in constructor is avaiable in all classes.



Inheritance 135

Inheritance has been used to add functionality to an exiting class. if a class B is derived by inheritance from a class A, we can say that B is a kind of A.

Levels of inheritance.

Classes can be derived from classes that are themselves drive.

```
Class A
{};
Class B: Public A
{}
Class C: Public B
{}
```

Multiple Inheritance.

A class can be derive from more than one base class is called multiple inheritance.

These terms signify the relationship between classes. These are the building block of object oriented 136

Association

· Association is a relationship between two Objects. In other words association defines the multiplicity between object

→

Aggregation :-

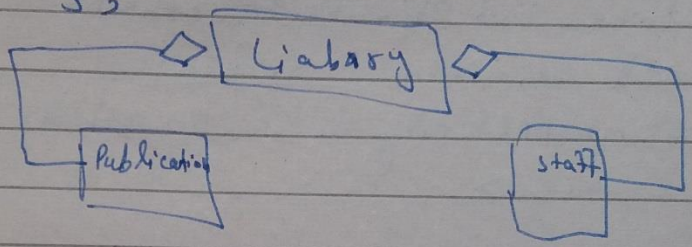
Both aggregation & inheritance are class relationship. that are more specialized than associations. it is instructive to compare & contrast them.

Aggregation is called has a relation ship. we say a library has a book or an invoice has a item line. it is also called Part whole relationship. The book is part of the library.

Aggregation is a special case of association

137

```
class A
{};
class B
{};
A objectA;
};
```



Composition 2-

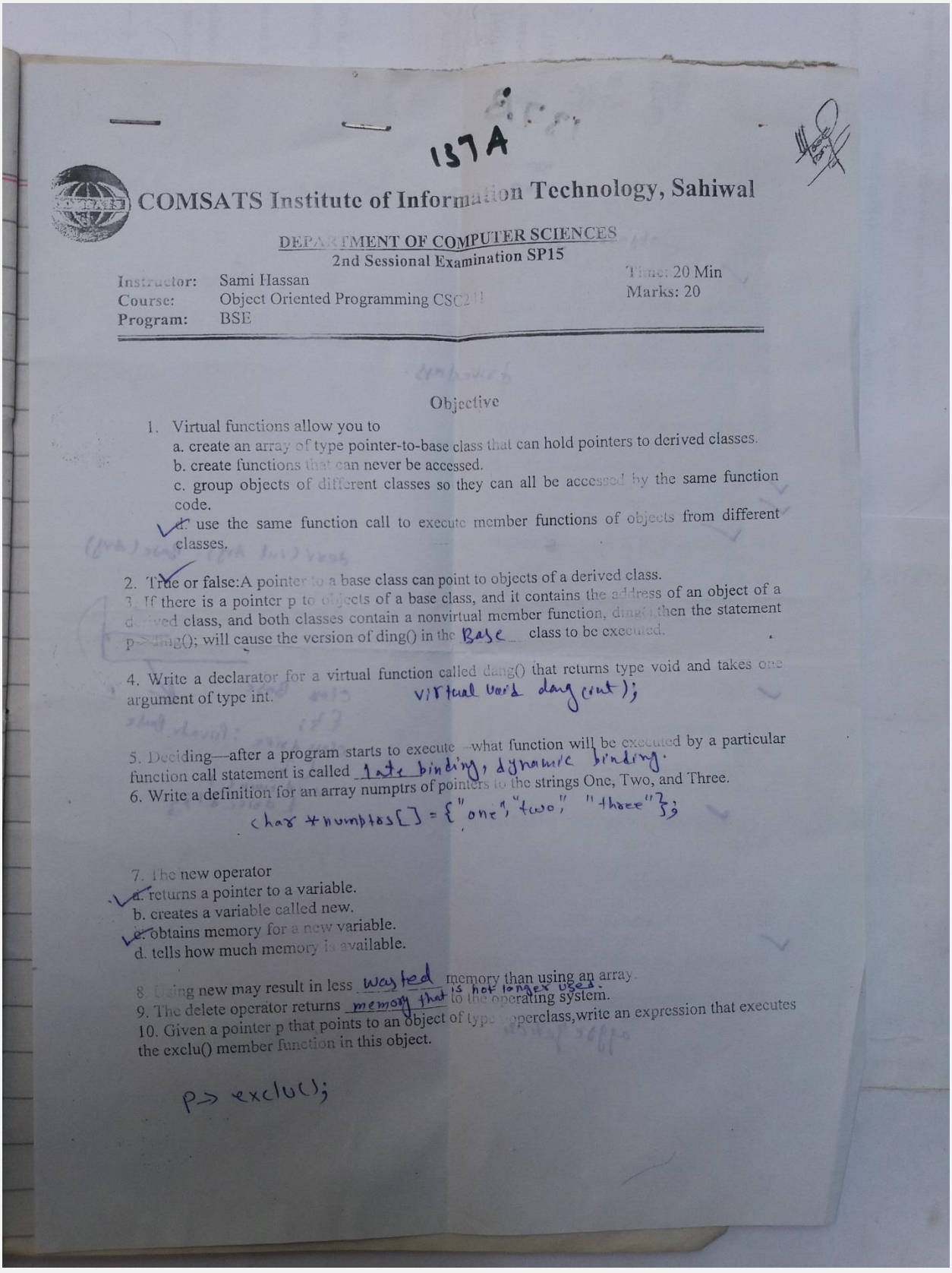
Composition is a special case of aggregation. In a more specific manner a restricted aggregation is called composition

it is a stronger form of aggregation with two addition.

① The Part ~~may~~ may belong to only one whole

② The lifetime of Part is the same as the life time of the whole.

A room is composed of a floor.



137A

COMSATS Institute of Information Technology, Sahiwal

DEPARTMENT OF COMPUTER SCIENCES

2nd Sessional Examination SP15

Instructor: Sami Hassan
Course: Object Oriented Programming CSC211
Program: BSE

Time: 20 Min
Marks: 20

Objective

1. Virtual functions allow you to
   a. create an array of type pointer-to-base class that can hold pointers to derived classes.
   b. create functions that can never be accessed.
   c. group objects of different classes so they can all be accessed by the same function code.
   d. use the same function call to execute member functions of objects from different classes.

2. True or false: A pointer to a base class can point to objects of a derived class.
3. If there is a pointer p to objects of a base class, and it contains the address of an object of a derived class, and both classes contain a nonvirtual member function, ding(), then the statement p->ding(); will cause the version of ding() in the Base class to be executed.
4. Write a declarator for a virtual function called dang() that returns type void and takes one argument of type int. `virtual void dang(int);`
5. Deciding—after a program starts to execute—what function will be executed by a particular function call statement is called late binding, dynamic binding.
6. Write a definition for an array numptrs of pointers to the strings One, Two, and Three.
   `char *numptrs[] = {"one", "two", "three"};`
7. The new operator
   a. returns a pointer to a variable.
   b. creates a variable called new.
   c. obtains memory for a new variable.
   d. tells how much memory is available.
8. Using new may result in less wasted memory than using an array.
9. The delete operator returns memory that is not longer used. to the operating system.
10. Given a pointer p that points to an object of type superclass, write an expression that executes the exclu() member function in this object.

`p-> exclu();`

137B

11. Given an object with index number 7 in array objarr, write an expression that executes the exclu() member function in this object.

objarr[7].exclu();

12. True or false: If no constructors are specified for a derived class, objects of the derived class will use the constructors in the base class.

13. If a base class and a derived class each include a member function with the same name, which member function will be called by an object of the derived class, assuming the scope-resolution operator is not used? derivedclass.

14. The scope-resolution operator usually
a. limits the visibility of variables to a certain function.
b. tells what base class a class is derived from.
c. specifies a particular class.
d. resolves ambiguities.

15. True or false: It is sometimes useful to specify a class from which no objects will ever be created.

16. Assume that there is a class Derv that is derived from a base class Base. Write the declarator for a derived class constructor that takes one argument and passes this argument along to the constructor in the base class.

derv(int Arg) : Base(Arg)

17. Assume a class Derv that is privately derived from class Base. An object of class Derv located in main() can access
a. public members of Derv.
b. protected members of Derv.
c. private members of Derv.
d. public members of Base.
e. protected members of Base.
f. private members of Base.

class Base
{};
class drive : Private Base
{};
void main()
{ drive d1;}

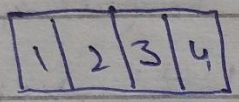
18. True or false: A class D can be derived from a class C, which is derived from a class B, which is derived from a class A.

19. A class hierarchy
a. shows the same relationships as an organization chart.
b. describes "has a" relationships.
c. describes "is a kind of" relationships.
d. shows the same relationships as a family tree.

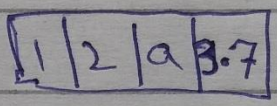
20. Composition is a _____ form of Aggregation.

aggregation

different in Array & Structer. 138



Array



Structure