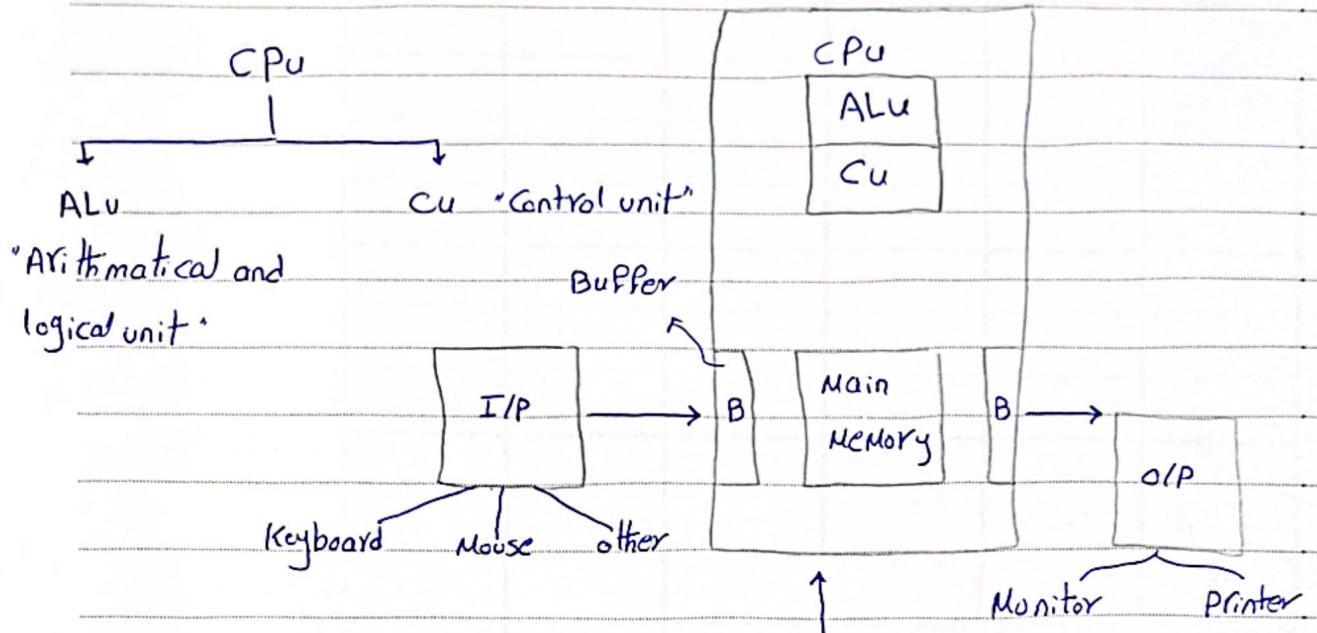


Introduction to programming using C++

"Day 1"

II How Computer work

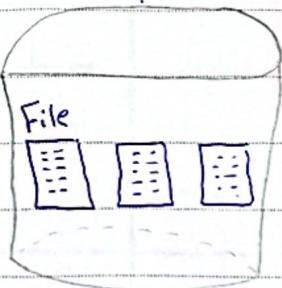


\rightarrow CPU
 Central processor unit
 Micro processor

يسود من مبنى اساس

III ALU

المسود من المعمليات - المسود من الحسابية
 arithmetic logical
 $+,-,*,/,\%$ $>,<,>=,<=,+=,!=, \&, \&$



Hard disk

E CU

هو المركب في الامدادات المحطة مسؤول عن تواجد

Ex I/P, O/P, main memory
hard disk

الخط

مع الامدادات المحطة مع CPU

* Hard Disk

پیغام کے لئے اس کا

types of File

Program File

ex NotePad.exe

Data File

.txt, .jpg, .mp4

* Main Memory

Main Memory میں کام کرنے والے Program کو

CPU کے لئے دینیا کو

electronic devices کو Computer کا CPU کو

(off/on, true/false, & low, high, 0/1, 0/1) یعنی

* Buffer

Data کا موقتی ذخیرہ کرنے والی

O/P ← Memory ← I/P || اسے

buffer

→ Numeric System

System

Base

Range

Binary

2

0, 1

Octal

0, 1, 2, 3, 4, 5, 6, 7

decimal

10

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Hexadecimal

16

0 → 9 & A → F

النهاية

$0 \rightarrow R-1$ ← Range اليمى Base → R اى تطبيق اليمى

فقط Binary system يفهم الـ Computer الـ

Ex 25 → 11001

لو سالب بيخرج خاتمة الاستارة ←

-25 → 111001 "Sign & Magnitude"
↑
Sign bit
↑
+ve -ve

→ +ve number #bits
→ Signed 0: 2 - 1
 #bit #bit - 1
 - 2 : 2

اى رقم (الكتير بيه سخزه كدا)

1.23 → 123×10^{-2}

ASCII ← Code يعطى English Letters الـ

a: 0110001 (97)

كل محرف 1 byte → ASCII الـ

⇒ what is Program ?

is to task على ما يملى Computer الـ

Program → Data + operation on data

Instruction

Programming languages يكتب بـ

الـ

→ Programming Languages

① Machine language →

```
0101001010  
1010110011  
0010110011  
1111001100
```

Machine language is a sequence of binary digits (0s and 1s) that the computer can directly execute.

Dis

Machine language is machine-specific and requires maintenance.

Maintenance is required.

② Assembly language →

```
load m1, r1  
load m2, r2  
add r1, r2  
move r1, m1
```

Assembly language is a low-level programming language.

Dis

Assembly language is machine-specific and requires maintenance.

Assembly language uses mnemonics to represent operations.

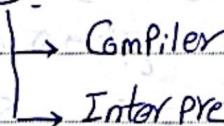
Assembly language is machine-specific.

③ High level language →

C, C++, Java, C#

High-level languages are easier to learn.

High-level languages are converted into machine code by compilers or interpreters.



Programmer → High level language → Assembly language → Machine language

الآن

Hybrid based - Wg

Interpreter / Compiler \rightarrow right \rightarrow 4

Compiler

Interpreter

Code ال يدخل ←
Machine Code ال يدخل
لو في Syntax error
(فتاد ال Compilation و هست
Program ال فتحت

line by line જીને જાણ.

statement الیمان دین

Machine Code لـ جوها

line کا سبق اور

۱۵۰

o/p File اور 15 ←
executable File ہو اک 10 ←
بیکملہ فرما و اجرا کرے گا
Compiler ہست ٹنائیں اور
Run کرے گا۔

کل نتیجہ کا error ہے

المحفلة ردوا

3 errors ای سکھنے کا

Run-time 31

Code 21

O/P File \leftarrow C:\file

line by line حرف حرف

ونقذة كل مرة هتل Yun

~~مترجم~~ Interpreter Code

ما هي لغات Compilers و ما هي لغات Machine Code

اداره و تلقیق Interpreter based

الطبعة

→ operating system

→ is an environment easy for a user to use the computer

Computer is a platform which can be used by users.

Ex windows, linux, Mac osx, Android

→ operating system talk all resource under control

and provide service

Computer is a platform which can be used by users.

RAM is part of OS which can be used by users.

resource is part of program which can be used by users.

Monitor or keyboard operation is controlled by OS.

OS will request for resources.

Access to disk or memory is controlled by OS.

hardware resources are provided by OS.

Method of OS to access hardware resources.

"System Call"

Program → API → OS

Application Program interface

Good

⇒ Programming Paradigm

① linear Programming

Stm1
Stm2
Stm3
Stm4
Stm5
Stm3
Stm4
Stm6
Stm7
Stm3
Stm4

sequential (لکود یعنی)

only one function (مكتوب في) "main"

Dis

يسهل مع المبرمج (1) (يسهل رفع و تغيير)

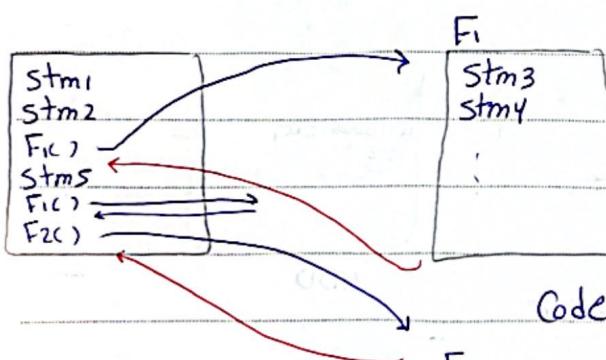
يسهل كرر (2) (يسهل حفظ كرر)

يسهل الصيانة (3) (يسهل رفع و تغيير)

يسهل الكسر (4) (يسهل إزالة الكسر)

② structure Programming

main



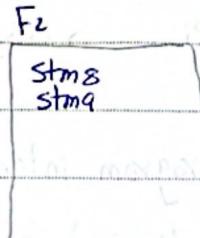
Adv

blocks (فقرات) (1) (يسهل رفع و تغيير)

task (مهام) (2) (يسهل رفع و تغيير)

debugging/maintenance (3) (يسهل رفع و تغيير)

Code reuseability (4) (يسهل رفع و تغيير)



Dis

global variable (متغير عالمي) (1) (يسهل رفع و تغيير)

restriction (قيود) (2) (يسهل رفع و تغيير)

no communication between functions (3) (يسهل رفع و تغيير)

الأسئلة

3] object oriented Programming

Algorithm is Function + Data + Encapsulate your logic

Adv

① Reusability الـ مـوـلـعـة

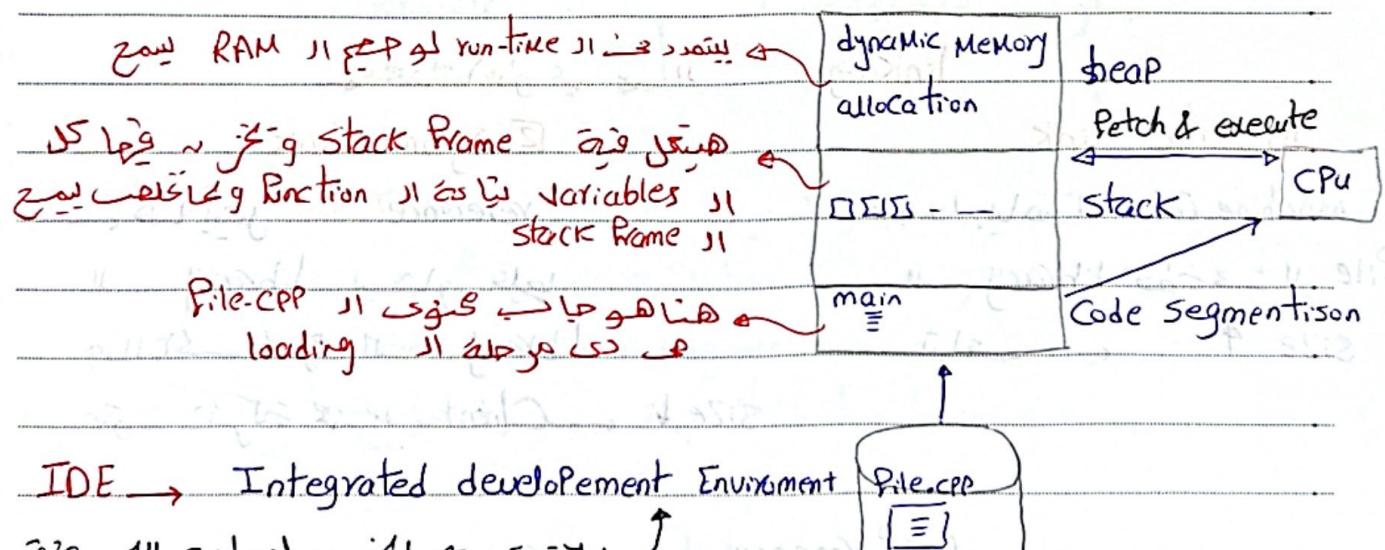
Abstraction

Data will be subject to restriction if required.

۲) maintenance فی اور Code ہے اور

→ Steps For Program Executions

- ١ Editing → Code مرحله كتابقه ای
 - ٢ Compilation → Machine Code مرحله ترجمه
 - ٣ Linking library → Program ای و dependence ای و library ای پربطي
 - ٤ loading → Code seg ای و HDP ای و Code ای و load ای
 - ٥ execution → Code ای و فقط ای و CPU ای



IDE → Integrated development Environment

file.cpp

三

HDD

— 1 —

二

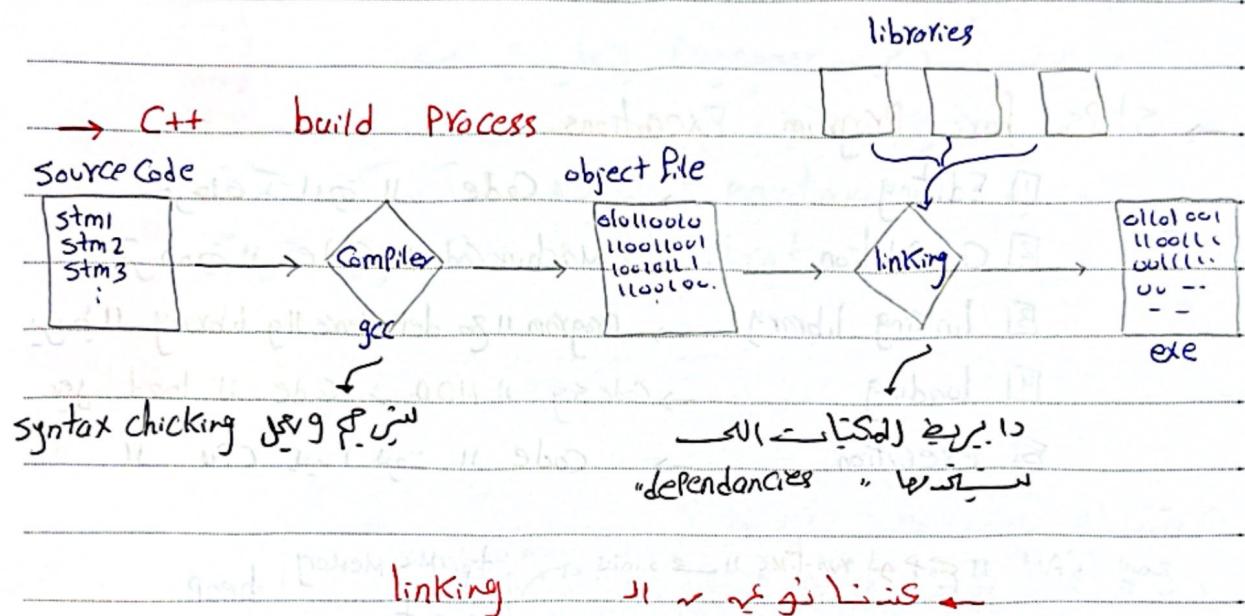
2020 RELEASE UNDER E.O. 14176

1

object File یا وسطی Code یا translate یا Compiler یا
Syntax error ممکن است

→ Debugging

logical errors یا Fix یا Code یا trace یا
tracing یا خطای کدی یا line یا break Point یا نقطه



linking

□ static link

Machine Code یا باینری

file یا static library

size ↑ تراویح

□ dynamic link

reference هستیج

library بیکار، علیجا

library یا پالسک یا

size ↓ Client یا ناچاری

Performance → same

الفیض

→ types of Errors

1 Compiler Errors

ای لکھ دی جو کو اس کو سمجھا کر کوئی نہیں کوئی دا
Syntax error

Ex Forget Semicolon ";"

→ Semantic Errors

meaning سب ملوات "Valid" ہو گا Syntax ہے کوئی

Ex $a+b;$ Valid Syntax

"Hello" / 2; invalid meaning

2 Linker Errors

→ The linker is having trouble linking all the object files together to create an executable

→ usually there is a library or object file that is missing



3 Run time Errors

پروگرام کے وقت میں ایسے ارور

→ Errors that occur when the program is executing

Crash یعنی program اور اس کی ساری

Errors یعنی کامپیوٹر کو معالج کرنے والے کاموں کا exception handling

Ex • Divide by zero

• File not found

• out of memory

کامپیوٹر کے کام کو کام کرنے والے کاموں کا

النها

4 logical Errors

output نتائجاً سخال یوں اے متنکل ہے بیطھعے Code اے ←
خطا

→ In Correct Algorithm

ڈا بکسٹریج اے ←

⇒ Compiler warning

Compiler ییدیلے کئیں ہے اے Code سخال عادی ←
warning ہے ایک لئنٹ چاچھے ممکن تؤثریں النقیہ کی کا فردی یادیں ←
عایا

وکیت وکیت دی warnings اے لمح بروہنگ اے Programmer اے ←
protective code

Ex int a;

Cout << a ;

warning "a" is used uninitialized

الافق


```

#include <iostream>           ← library built-in
using namespace std;
int main() {
    Starting Point           ←
    cout << "hello world!" << endl;
    return 0;                ← return to os
    Console output

```

مقدمة بسيطة في main() وهي المدخلة الوحيدة في البرنامج ←
Program ←
Program هي المدخلة

→ namespaces

ما هي namespaces أو class أو functions التي نستخدمها في البرنامج ←

يسعى عالميدها إلى تنظيف واقتراضها ←

ويتم إدخال namespaces في نفس class أو في ←

فهرس الأشياء التي نريد الوصول إليها ← namespaces

include ←
namespace ←
namespace

① Explicit namespace

std:: ←
النطاق

② namespace directive

using namespace ns; ←
استخدام

③ Specific thing

using namespace std; ←
النطاق

و دا حامضه دا
namespaces ← دا حامضه دا

std:: ns ← دا حامضه دا ←
النطاق

→ I/O on Console using cin, cout

Cout << value;

insertion operator
output stream

Cin >> variable

extraction operator
input stream

endl or "\n" → new line

good

→ Variables

هو اسم تطلق على مemory location هو اسم تطلق على مemory location

→ Variable value may change

① Declaring variable

Datatype variablename; → ~~garbage data~~

② Initializing variable

Datatype Name = value; Datatype Name(...), Datatype Name { ... };

Constructor initialization

C++ syntax
range (check جزء
of values) (values)
error (error)

→ to get Address Any Variable
& Name

and operator ↑

⇒ Identifiers Naming rules

① Can Contains letters, numbers, and underscore

② Must begin with a letter OR underscore "_"

③ Can't begin with a number

④ Can't use reserved keywords

⑤ Can't redifined a name in the same scope

Ques

→ Best Practice Naming

① Be Consistent with your naming Conventions

→ using Camel Case "myVariable"

→ " Snake " "my_variable"

→ " Pascal Case " "MyVariable"

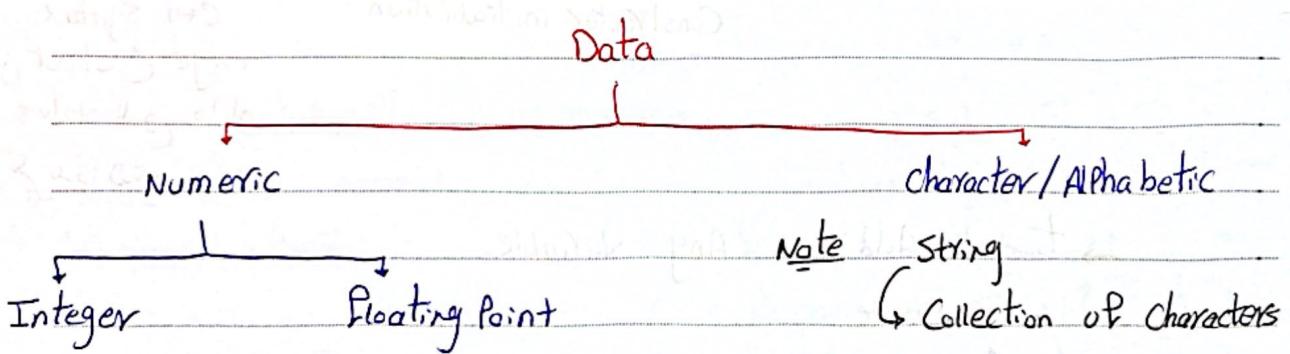
② Avoid begining names with underscores

③ use meaningful names

④ Never use variables before initializing them

⑤ Declare variables close to when you need them in your code

⇒ Data types



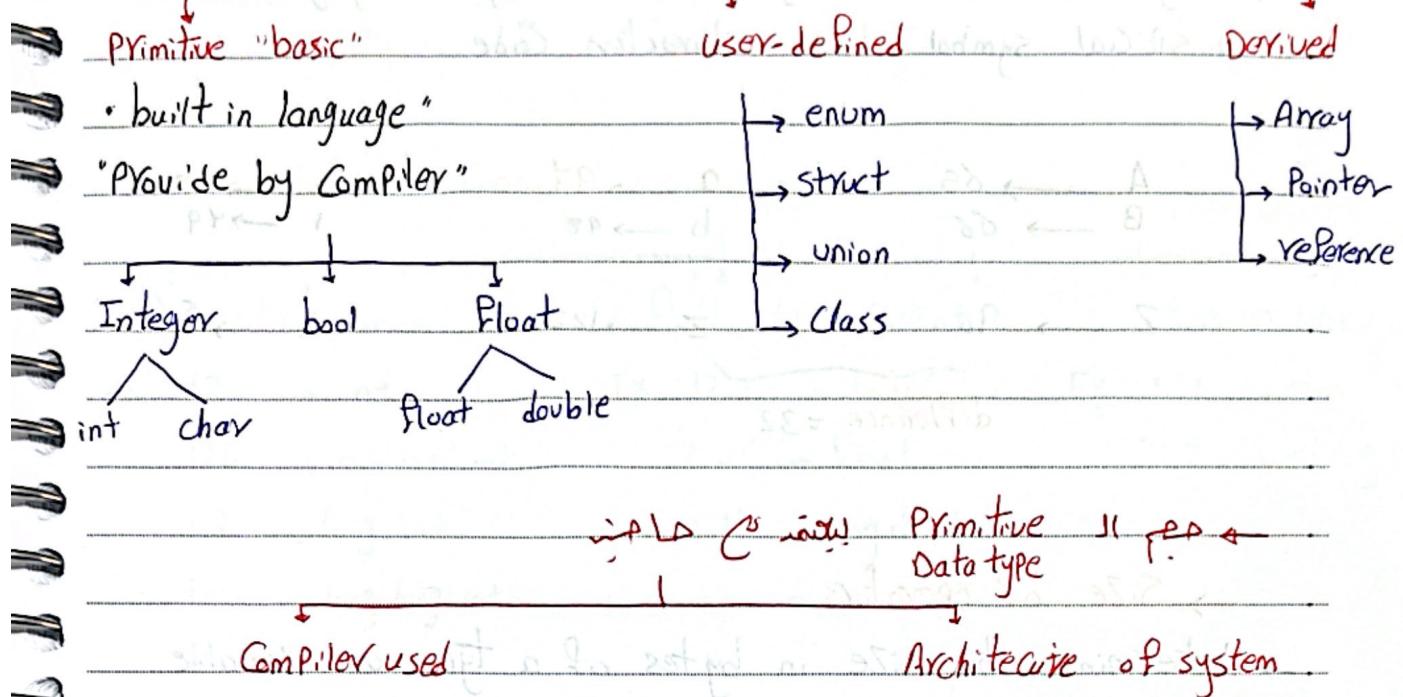
الذى يخزن فى ذاكرة الـ memory Data types

ويمكن انجذابه على عمليات operations

الذى يسمح به الفاصل Range

الآن

Data types



"Address bus" Architecture of system

32-bit 64-bit

→ Modifiers

char, Int	unsigned	2^n
		-2^n to $2^n - 1$
long	Signed	2^n
		-2^n to $2^n - 1$

Data type range

~~long~~

short

Data type range

long char → invalid

ASCII code is just 1 byte $\Rightarrow C++ \rightarrow \text{char}$

Good

→ ASCII

→ every alphabet or every letter in english language as well as a special symbol have character code

A → 65
B → 66
⋮
Z → 90

a → 97
b → 98
⋮
z → 122

0 → 48
1 → 49
⋮
9 → 57

difference = 32

→ Size of operators

→ determine the size in bytes of a type or variable

Ex `sizeof (int);`
 `sizeof (x);`
 `sizeof x, y, z;`

→ limits & cfloats

→ Contain size and precision information about your implementation of C++

Ex `#include <climits>`

INT_MAX

INT_MIN

LONG_MIN

LONG_MAX

`#include <cfloats>`

FLT_MIN

FLT_MAX

good

⇒ Constant

→ Value Can't Change once declared

→ types of Constant

① Literal Constant

Integer literal Constant

Ex 12 → int

12U → unsigned int

12L → long int

12LL → long long int

Float literal Constant

Ex 12.1 → double

12.F → float

12.L → long double

Character literal

Ex ' ' → char

" " → string

Escape Sequence

\n → new line

\r → return

\t → tab

\b → back space

\' → Single quotes

\" → double quotes

\\" → backslash

② Constant declared using "Const" Keyword

Const type Name; → if local variable "rabage data".

Const type Name = value;

Const Datatype Cons var = value;

Ques

③ Enumerated Constant

```
enum Name {  
    VarName, ---;  
};
```

لیڈی لے \leftarrow
value \leftarrow
Constant \leftarrow enum \leftarrow قیمت اے
معروضت لعنہ \leftarrow

لوہت عامل واپس بھی دھین \leftarrow

کل واحد یا جو فتحہ الک فیلہ وین ود (۱) \leftarrow

Ex

```
enum dep { CS = 1, ECE = 2, 2+1=3 یعنی  $\leftarrow$  it };  
enum day { Mon = 1, Tue = 2, wed = 3, Thur = 4, 5+1=5 یعنی  $\leftarrow$  it };  
day d = mon;
```

dep d;

Discrete values لیے جو enum لیے

d = CS; readability کے لیے program میں "int" وظیفہ

④ defined Constant

```
#define Name value  
replace new compilation by preprocessor  
کے حامیہ بارے کے دا سارے دیے جاوے یا کسی دیے جاوے کے  
enum gl Const دریے memory لیے
```

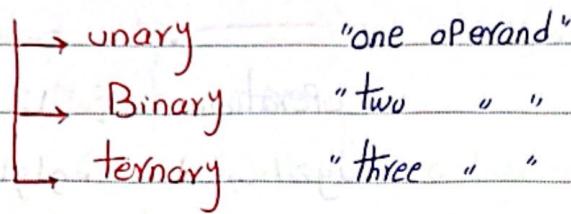
→ **typedef** "alias name to datatype"

Ex **typedef** int i; Performance لیے جو

i x = 4;

الآن

⇒ operators



1) Arithmetic operators

$+, -, \times, /, \%$ int مجموعه بیتی

2) Relational operators

$<, >, \leq, \geq, ==, !=$

3) Logical operators "short circuit"

$\&\&, ||, !$

4) Increment, decrement

$++, --$

"Prefix" طبق الاول "Prefix" لینک داده

"-, + بیتی Value" Postfix لینک داده

Expression operators جملہ بنانے کے لئے

Note $x++--;$ "undefined"

5) Bitwise operators "bits کو سمجھنے کے لئے"

$\&, |, ^, \sim, \ll, \gg$

left shift → right shift

$x \ll i \rightarrow x * 2^i$

$x \gg i \rightarrow \frac{x}{2^i}$

6) Assignment operators

$=, +=, -=, *=, /=, \% =$

left hand side = right hand side

اللهم

⇒ Associativity and Precedence

precedence اول نفس اول operations
left → right ~ نسبی
right → left

نقولنا اربع operation لتفتيت الچوبل

→ ↗ Associativity & Precedence →
rules to determine the order of operations in an expression

Notes

$x++$;  PostFix

`++x;` → Prefix

$$\sum_{i=1}^n x_i = 2, y_1$$

$$Y = ++x; \rightarrow \textcircled{1} x = x + 1 = 3$$

$$Cout \ll x; \quad 3 \quad \textcircled{2} \quad y = x = 3$$

```
Cout << y; 3
```

→ if operands are of different types will casting

→ IMPLICIT Casting

`int` → `unsigned int` → `long` → `unsigned long` → `float` → `double` → `long double`

→ داعم اور type اور صورت یعنی types ریکر

short & char always converted to int

الآفاق

→ Explicit Casting

→ ~~not~~ type → type ~ موارد لوجيقيات

① Promotion

↳ Conversion to a higher type

② Demotion

↳ Conversion to a lower type

→ Casting Syntax

① (new type) variable; ↗ loss مسأله ← المدى

② static_cast<new-type>(variable);

Cast after variable || as check الجواب

→ Data || loss مسأله

Good

Day 2

⇒ Conditional statement

if (Condition) {

لوكفت الموجة هستة دا

g

else {

لوكفت الموجة هستة دا

g

Nested if \rightarrow if و else if () \rightarrow نفس كما نعم

switch Cases \rightarrow " == " equality عبارة Condition \rightarrow لوكفت الموجة هستة دا

switch cases \rightarrow if \rightarrow switch \rightarrow نفس

switch unique

switch ()

variable (int, long, short, char)

case : \rightarrow label

only

" خواه "

break;

literal " Constant " EX ' A ' - 10

default: \rightarrow نفس

each case Must end by break

if all Case pull throw \rightarrow to avoid logical error

Case \rightarrow زياد مطولة g هستة على توقيع \rightarrow switch \rightarrow لوكفت الموجة هستة دا

switch \rightarrow عالي بالدور ، لذلك ارجو check if \rightarrow نفس

limited \rightarrow jump

فقط int, char, go

الافق

Case grouping \rightarrow If Pull throw \rightarrow (فرو اسپیس) +
break; اجتنب \downarrow

Switch () { Cases } group \rightarrow تبلیغ

case _____ : } \rightarrow الک لها نفس او O/P

case _____ : } \rightarrow Pull throw \rightarrow Case If \rightarrow مجموعه \rightarrow المجموعه

(cases) If Code \rightarrow repetition \rightarrow (عمل) \rightarrow Pull throw \rightarrow اسفل سینه

\Rightarrow Conditional operator "ternary operator"

(Condition Expression) ? expression 1 : expression 2;

true \rightarrow دالو \rightarrow false \rightarrow دالو \rightarrow Pake دالو

\Rightarrow looping / repeating statement / iterative statement

بنابراین دالو همچنان که در ورایتی \rightarrow

کتابت شده است دالو همچنان که در Function \rightarrow

Ex Cout << "hi"; \rightarrow دالو hi اسکرپت دالو

Cout << "hi"; \rightarrow دالو کارکردن دالو

Cout << "hi"; \rightarrow دالو کارکردن دالو دالو loop

Adv ① reduce repeated code

② Maintenance easy

النهاية

کی اتفاق Convention لیں۔

For	while	do ... while
دی لمعرفت	دی نفضل لنفذ	لو عازم (نقد)
کد مراتب تکرار کوڈ	کدوں ها ال Condition	ماجنز هرہ واردہ
→ need Counter	متلقت	سچ اساق
For (initialize; condition; step) {	while (Condition) {	do {
≡	≡	≡
g	}	}
مخصوصات - الشند		g while (Cond.)
initialize ①	true بدلے اور لے	کو اسیق اور ب
کد، عراست المزرات + 1	لتفہ کو False بطلع	Check کو بدلے
check ②	Code اسیق true دو	Code اسیق
Code ③		نای 1509
Step ④		در مراتب تکرار کوڈ

int i; for (i = 0; i < n; i++) {
 cout << arr[i] << " ";
}

For $C_i=0$; —; —; —; —
 $y \equiv$ For II step check IIg step (اعویں اور

لیں کے بیچ میں ایک سیوف الکوہ بھریٰ ہے جو اسی کا اول سفر

Note `foreach` → index \rightsquigarrow element یک دفعہ دیکھ کر by element

For (Datatype Name : VariableCollection) {

لهم نسخ بحث المذاهب ستة عشر Collection

لو هست عاروفاً يكلاها

Ex ↗ Array

الآباء

Continue → next iteration used in loop

break → terminate loop " " → switch.

loop

Stack overflow مسح infinite loop لوگوی

Notes

`<conio.h>` { `getch()` → Enter ناچار حرف هست بسته بپرسی
 ناچار حرف هست بسته بپرسی از کارته
 `getchc()` → از کارته ناچار حرف و از کارته

`Gn >> — ;` → space gl enter بخوبی

`system("cls");` to clear screen "cmd"

ای کد دستیت بینکت بینکت داشت
سحل آنکه ای ای دلخواه تغیر کرد من ای سطر

`textattr(ox_)` → Function کد

لو_ الخلفية
0 → black
7 → white

دلخواه کانهالنا
جا هم نزدیک

`Gotoxy(int x, int y)` Console ای Cursor دی نیز کار کند.

Good

Day 3

→ Array

Must homogenous "Same datatype"
"Physical meaning"

temperature & age مثلاً مفهوم موحد

Datatype Name [] ; \rightarrow Size = #element

↳ No checking boundary

Memory It is collection of elements

Ex int arr[3];

4 byte

Size array = $3 \times 4 = 12$ byte

arr →	0 arr[0]	0x1000
	1 arr[1]	0x1004
	2 arr[2]	0x1008

↳ element always has pointer to array itself

→ Array Fixed size, Can't change size in run-time

→ int arr[3] = {1, 2};

↳ element 3L جملہ

1	1	0
---	---	---

→ int arr[3] = {0};

↳ element 3L جملہ

0	0	0
---	---	---

→ int arr[3] = {-1};

-1	0	0
----	---	---

Good

`int arr[3] = {1, 2, 3};`

`arr[7];` *"Not compilation error"*

{

undetermined behaviour

`arr` → address first element

`arr[7]` → value of arr

Crash

يختل عادي

الخطأ في الـ

لو عقلي في الـ

out of Access

→ `int arr[3] = {1, 2, 3};`

↓ *length or size*

out of bounds هو المفترض أن يكون

1D Array → need one index to access

2D Array → " two " "

Matrix

"row, Col"

→ Multi Dimension Array

Data type Name[][];

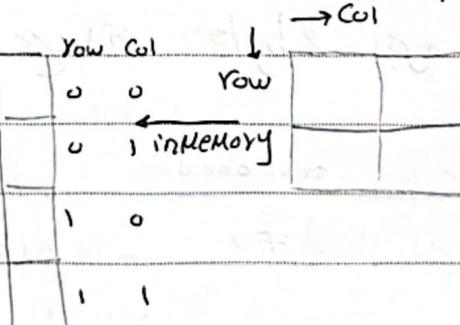
row Col

Row X Col = Element No's

فقط 1, 2, 3 في memory

1, 2, 3

EX: `int arr[2][2];`



`int arr[3][4] = {{0}, {1}, {2}}; → all element = zero`

→ 3D `int arr[][][];` array to 3D array

Good

Normal Key → لینک اسیکی کی دس کے لئے

1 byte پر 8 bit 127 char پر 8 bit

Extended Key → 2 byte

Ex UP, Down

Extended key ← -32
First byte Second byte
→ Ascii Code of char
-32 ≠ Normal key

getch() → Extended key لیں
" -32 ہے " First byte
" Ascii Code " second

Keyboard → buffer
Contain only one byte
buffer پر پڑھنے کے لئے
pFlush(stdin)

یعنی 2 byte کے char کی تعداد کو ↪

↪ 0x15 extended key

0000 0000

#0

normal key

extended

گزینہ

Day 4

→ String

→ Array of char

char name[6] = "ahmed";

$10 - 1 = 9$ متاع لی عارف

کوپی کر کرے Compiler کا تو

String میں دھایا تو '0' آ جائے

a	0
h	1
m	2
e	3
d	4
l	5
o	6
0	7
0	8
0	9

→ Can initialize like

char name[n] = { 'a', 'h', 'm', 'e', 'd', 'l', '0' };

" " [n] = " " ;

Valid n. of char = n-1

Cout << name; → پڑھنے کا موقع
" " و "0" کا موقع

→ Can Access by index

name[idx];

' ' → char literal

" " → string

Note int name[] = "ali\0 sameh";

() [a] [l] [i] [] [0] [s] [a] [m] [e] [h] تھیں تو

index 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 ← unreachable by name

→ once string created can't use " " again

char name[5] = "ali"; → اکونا الوجہ کو لے لیں

name = "ahm"; → Not valid اس کو لے لیں تو

واعزم کرو

Good

"Enter gl space" لیفظ نظر کو ادا کرے Cin اور
"get string" getsc(); سترینگ کو String میں (فر) عدے جائے VariableName

size اور کسی String کو دخل کردار معلوم نہیں
undetermined behaviour

to calculate length any array

$$= \text{Sizeof}(\text{arr}) - \text{Sizeof}(\text{arr}[0])$$

⇒ built-in Function

<string.h>

① strlen() → "lo" کو الحروف یہوں

② strcpy(destination, source) کسی destination = source

کو اس سیو بیٹا وروں میں String ہے اسی کا deep copy کیا جائے

③ strcat(dest, source)

dest کو اسی String کو جو پڑے۔

④ strncat(dest, source, n)

نچھے

⑤ strncpy(dest, source, n) کو n کو copy کیا جائے

⑥ strstr(,)

one ہے اس کو String ہے اس کو String کو

one کو اس کو string کو جو دیکھے جائے

⑦ strchr(str, char)

one اس کو اس کو substring کو دیکھے جائے

Good

⑧ strcmp(str1, str2) two string میانیں

کوئی نہ ایسے پڑھ پر بینے کرو

$str1 - str2$

return

↓

-ve

zero

+ve

$str1 < str2$

$str = str2$

$str1 > str2$

A ≠ a تو is کے ignore Case جو کو $strcmp()$ نہیں

وہ " " کے $strcmp()$ نہیں

String ایسے عامل کو عبارت کو

نہیں گوئی کے اسے اسے جو کہ

2D Array پریمیو

char names[][] = { " ", " ", " ", " " };

Strings ہیں ↗ ↗ string کو گرفتار کریں

⇒ Character Functions <cctype>

① isalpha('')

② isalpha() isalnum()

③ isdigit()

④ islower()

⑤ isprint()

⑥ ispunct()

⑦ isupper()

⑧ isspace()

① tolower('')

② toupper('')

گرفتار

ویلے

Good

→ String using class `<string>`

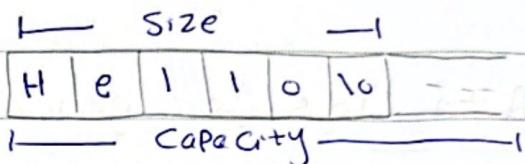
`String name = "ali";`

`name = "ahmed";` → (5 write `ahmed`)

`getline (cin, name);` → spaces (مفردة سفر كامل بـ)

EX

`String str = "Hello";`



① `length()` ≠ `size()` "المحفوظة ليوفر

② `capacity()` "السعة المحددة للبيانات

③ `resize(n)` "السعة المحددة للبيانات

depending on Compiler ← ④ `max_size()` "رس Capacity" رفع السعة

⑤ `clear()` "مح او حفظ" (مح او حفظ)

⑥ `empty()` "غير ملئ" (غير ملئ)

⇒ operator of string class

① `at(idx)` ↪ `[idx]`

② `front()` → ترميم اول حرف

③ `back()` → .. (آخر)

④ `+` → Concatenate

⑤ `=` → Copy Content

replace, insert, append is, Class is built-in function

الآن

→ A Functions

Linear Programming

→ repeat code

→ Maintenance ↓ "Complex"

Structural Programming

blocks اکی Code ہے

Adv

① Maintenance easy

② avoid repeated code

block of Code نوشتار نفس الـ

سادیتی Function

Dis

→ can't trace and data validate of global variable

loop، نفس ال statements و، بعض تكرر ال

عن اهالك هنفرونة

leg call relaying Functions position

return type identifier Name () {

main II ja declaration jisr pjs

returntype IdentifierName()

prototype / header

هذا يسمى مترادفات  \rightarrow مترادفات \rightarrow Parametrical \rightarrow متغيرات

الْأَقْدَمُ

→ Calling

IdentifierName();

argument

اک کے جو ادا Code اور jump جیسی
Call کا حصل فتنہ call
ویسے یعنی یکل میں call کا حکم

Note

لو اور فہرست Parameters کو function

Compiler کی argument کو ادھر تک call کرنے والا عمل error ہے

EX void Hi();

cout << "Hi";

g

لو جملہ کا
error ہے

int main() {

Hi("ali");

g

کوئی
error ہے Compiler کی

error

Formal Parameters

→ Parameters

function کے لیے اور Data کے

→ Arguments

" " کا Call جی " " " "

actual parameters

→ Solid

S → Single Probability

اپنے task کو یہ عمل و کامیابی واحد کے لیے حل کرنا

task کی ایک task کا یہ ساتھی میں اس کا دھنہ نہیں اور

الزیادہ کم اور

النفع

وہر آئندہ زکر میں Function کا return دا جزو اسی میں ہے۔
وہ only one value ہے۔

Ex

```
* int add(int x, int y) {
    return x+y;
    return x-y; → valid but unreachable code
}
```

error ہے اسی Compiler کی نسبت میں سمجھ لیا جائے۔

Note

```
Void Hi() {
    cout << "Hi";
    return; → valid → Function کا بھی جزو ہے
}
```

Note

Stack	→	Size ↓	Faster ↑
Heap	→	↑	" ↓

Parameters کو developer نے کہا تھا default Argument ہے۔
"list کا پہلی default اسی" default کی کوئی ہمیشہ Argument ہے۔

Ex int add (int x, int y, int Z=10, int a=20) {
 return x+y+Z+a;

→ Calling add(10, 20) → 10+20+10+20 = 60

add(10, 20, 30) → 10+20+30+20 = 80

add(10, 20, 30, 40) → 10+20+30+40 = 100

Goal

⇒ Reference Variable

memory جاگہ کے لئے alias نامی جو نام ←
Datatype & Name = variable;

declaration انتدا initialize پر جو ①

alias کے لئے اعماق میں رجسٹر ②

memory جاگہ کے لئے جو ③

Ex int main()

int x = 10;

X/Y

int &y = x;

X/11

x++;

0x1000

cout << y << endl; → 11

}

[لہجہ اس کا] "X" است 0x1000 کو اسکا نام کہا جائے اس کا ←

int x=10; 0x1000 است X

reference places کے alias name کے لئے لفڑی دیں کہا جائے

int &y = x; y کو 0x1000 کا alias name ہے Y کا

X کا نام کہا جائے

0x1000 کا alias name ہے X, Y کا 15

یہ Y کا X کا address ہے

memory میں بچھن لئے ہوئے reference کا ←

alias name کا table ہے

address کا call by ref کیا جاتا ہے ←

(پڑھو) Space بچھن alias name کا ہے

الآن

⇒ Calling types

① Call by value

"Copy" Data جاہے ایسے کسی ←
original Data جاہے Copy Data جاہے تھیں کسی ←

```
int main() {
```

```
    int a=10, b=20;
```

```
    swap(a, b);
```

```
void swap(int x, int y) {
```

```
    int temp = x; } local variable
```

```
x = y;
```

```
y = x;
```

لئے ایسا جاہے!

~~temp = 10
x = 20
y = 10~~

~~a = 10
b = 20~~

Swap Frame

main stack "stack frame"

Stack

لئے ایسا جاہے!

کہاں main جاہے

② Call by address

original value اور یا pointer کو اس کو اس کا هدایت کروانے کا ←

original value کو اس کو جو کرے ←

```
int main() {
```

```
    int a=10, b=20;
```

```
    swap(&a, &b);
```

~~x = 0x1000
y = 0x1004
temp = 10~~

Swap Stack

~~a = 10: 0x1000
b = 20: 0x1004~~

main stack

```
void swap(int *x, int *y) {
```

```
    int temp = *x;
```

```
*x = *y;
```

```
*y = temp;
```

Q

3] Call by reference

فی ال کال بای رفرینس کیا ہے اعلیٰ تر ہیجست فی ل الجل اور
Space میں Copy ویا ملٹیپل ڈیوائیس (عمل کا دیوائیس) میں
Call by reference ہے

Ex int main() {

int a=10, b=20;

swap(~~a~~, a, b);

y

Void swap(int&x, int &y){

int temp=x;

x=y;

y=temp;

y

alias
Name

a, b

Call by

Data جل

لو عارضہ میزبان

Const مختیہ

Const int &x = y;

x=90;

Error → Data جل میں جو

Reference ہے

① Variable ای

① scope

المکان الکھو دیا

دو رہ جیا دیا جائے اور

وجہ واقع اسکھو

بیکاری دیکھو (میں)

میں فی Scope کی وجہ (قر ایک فیکھو اسکے نیکے اس سر)

& scope ایں ایں

التفہی

⇒ Global Variables

متغير (غير ا Temporary) اي مكتوب وال
Code لا يخفيه او run لا يخفيه

ای Function \leftarrow متغير (Temporary) اي مكتوب لها هتليب
local \rightarrow global او

داخل او \leftarrow Function \rightarrow نفس local variable او اي Function

(::) scope resolution operator \rightarrow global variables \rightarrow لو هست کے

Ex int g = 10;

int main() {

int g = 9;

namespaceName:: varName

Cout << g; // 9 NAME \rightarrow لو هست کے نام

Cout << ::g; // 10 SPACE \rightarrow هست کے دا

lifetime \rightarrow Stack او \rightarrow global variable \rightarrow دا

⇒ static variable

هو متغير ينشر في كل مرة و اعادة فقط مع اول مرة

فقط ولو سلت من فعل كل مرة موجود

scope \rightarrow فقط Function \rightarrow دا

lifetime \rightarrow Program \rightarrow Function \rightarrow call اول

• stack \rightarrow " " \rightarrow دا

datatype static Name = value;

conclusion

⇒ Passing Array to Function

return type Name (Datatype name[], int n) {
≡ size مساحت ای
هذا size size
array ای خواهد بود
original Data ای خواهد بود
Pass by address بخواهد
→ arr Name (arr); array ای خواهد بود
که element ای خواهد بود

? Pass by Address یعنی Array ای خواهد بود
array & Copy همچنان Pass by value نه
Pass by address که فاصله بین شیوه کار کردن میگیرد

⇒ Return Array from Function

local درینجا بینیم Function → Array میگردید
النک که شیوه اولیه و بالاتر که شیوه دیگر است

Pointer ای خواهد بود لے کر ای خواهد بود

```
int[] Name() {  
    int arr[3] = {1, 2, 3};  
    return arr;  
}
```

ویلی

⇒ Struct

Value \rightsquigarrow معرفہ کرنا user defined datatype اسے سمجھا
 "Composite Data" کاٹھے datatype ہے

```
struct identifier {           Member
    Datatype Name;
}
```

→ to Access element "member" in struct

var Name . memberName
 dot operator

→ Size of any struct = Sum (Size of each member in struct)

Ex Struct emp {
 int id; } بیکاروں
 char name[20]; } memory میں کہاں
 g;

Void main() {
 emp emp1; } initialize جعل
 emp1.name = "ay"; X error
 strcpy(emp1.name, "ay"); ✓
 g

Good

X arr = arr2 IS two Array J Assign Jas) arr2
arr (غير (غير كا مع arr سبب انتم يخوا في نفس النوع
emp emp2 = emp1; V → deep copy
values هي اخر سطوة من arr

memory J1 is address of struct gg's variable ↗
" & " p1 is L-loc-كادس لـ ↗
 $\& \text{empl} \rightarrow$ struct J1 is member Jgl address ↗

Value equality \neq Struct JI ↪

Ex `empl == emp2` ↗
is \neq values \rightarrow `True`

• تعریف اد Struct بکو ~ فیلم آندر گرین

local user defined datatype function دا جل ای struct دا جو را کرو
جسے (کو را جو ممکن نہیں ہے) scope دا جل ای variable دی وجہ پر Function دی وجہ

→ Pass struct to Function

Returntype Name (structName vorName) {

三 Pass by value, ref struct

array of جملہ ←
struct

Ex emp emp[3] = {{2, "ali"}, {3, "khalid"}, {5, "Said"}};

→ to Access

Cout << emp[1]. id; // 3

id : 2 emp[0]

name: ali

Cout << emp[1]. name; // khalid

id=3 emp[1]

name: "khalid"

id=5 emp[2]

name: "Said"

one's cell number of struct variables initialize کیا گی

Data type as default value کیا گی

Ex int → 0 string = "lo"

char → "

Ques

Day 5

→ Pointer

address بخزنه variable هو

int $x = 10;$

الخواص table يسمى OS أو

"mapping" 10 في x است

byte by byte وين المكان

x [10] 0x1000

↓

↓

لو عايز متى في الخواص فتح الخواص

Pointer لـ ترجمة

int * ptr = &x;

نوع البيانات

ptr [0x1000]

العنوان على

x بخزنه

Cout << x; // 10

Cout << ptr; // 0x1000

Cout << *ptr; // 10

indirect operator

يقوله نوع البيانات التي يشير لها ptr

int x معرفة ptr بـ 4 byte كـ Data واقع

double d = 2.3;

double * ptrd = &d;

d [2.3] 0x1004
ptrd [0x1004]

8 byte → d المكان الذي يحتوي على

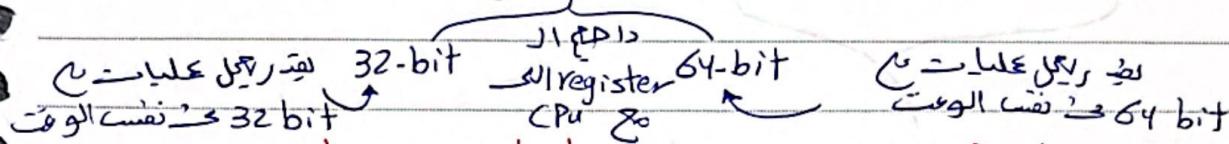
البيانات المعرفة السابقة

الآن

الک Data type تابت خاندیم نوع ار Pointer ده

کمپیوٹر کی bit میڈیا کی system پر بستھے ہیں اسی پر اسی پر علیها ہے

"memory" ای مکان فن ار address یوچل لے



کمپیوٹر کی Data ار byte جو یخزنا گواں اول Pointer

لے سکوں علیها

Note

`int * ptr;` → *garbage data*

undetermined behaviour → *garbage address*

ستخوابی

crash

لوالکرست

لوار *garbage address*

مسکونی

فسخی

او معوی Program

لیکن

معناد (و مدل نیاں اکاچھے الک نیتاو، علیها

* "indirect operator"

→ to declare multiple pointer in one line

`int * ptx = &x, *pty = &y;`

`ptx = pty;` "address equality"

ایسا نیتہ بیٹھوں نفس اکاچھے

Note

`int * (ptr, pty);`

Pointers

Pointer کا ادا

کوئی

operation on Pointer

↓
addition

↓ subtraction

↓ No multiplication & division

مقدار مرفق (جمع او اطرح عن اشاره)

مقدار مرفق (جهاز الجمع او النقصان)

step تقول على int number

Ex int $x = 6;$ x

	0x1000
	0x1004

 int * $P = \&x;$

$P = P + 1;$

unit step دعه تقول على address

address دعه تقول على address

step int دعه تقول على 4 byte

4 byte

so Datatype of $P = \&x;$

$P = P + i; \rightarrow P = P + (\text{size of(Datatype)} * i)$

original data دعه تقول على Function

Function دعه تقول على address

struct دعه تقول على struct
 ↓
 ① struct ② Pointer
 ↓
 value دعه تقول على value
 return دعه تقول على return

work around solution

↓
 دعه تقول على

الافق

Stack میں static اور Array میں جو Function ہے۔
Heap پر جو دینا ہے dynamic ہے۔

⇒ Pointers & Array

`int arr[5] = {10, 20, 30, 40, 50};`

`int * ptr = &arr[0];`

`Cout << *ptr;` → 10

`Cout << *(ptr+1);` → 20

`Cout << *(ptr+2);` → 30

`Cout << *(ptr+3);` → 40

`Cout << *(ptr+4);` → 50

arr 0x1000

ptr 0x1000

array کو
pointer کو

10 arr[0] → 0x1000

20 arr[1] → 0x1004

30 arr[2] → 0x1008

40 arr[3] → 0x1012

50 arr[4] → 0x1016

$ptr + 1 \rightarrow 1000 + (1 \times 4) = 1004$

$\therefore ptr + n \rightarrow ptr + (n \times \text{sizeof(Datatype ptr)})$

array کو
پسیں فرما

* (arr+1) 15

[] indexing کیا ہے پسیں [,] array کے، جو Pointer ہے۔

Ex `int * p = &arr;`

`Cout << p[1];` → * (p+1)

arr کو all address کے جو Const Pointer ہے array کو فرمائیں۔

`int x = 10;`

واحدی نامہ کو، 3، 2، 1 کو نامہ کو

`int arr[3] = {1, 2, 3};`

`arr = &x;` X invalid

Good

ا) Variable Pointer و ابزاری Pointer ۱۱ ←
ب) Const Pointer ای جزوی ای تابعی

```
int x = 10;  
int arr[3] = {1, 2, 3};  
int * p = arr; ✓  
p = &x; ✓  
arr = &x; X → incompatible type
```

array ای قدرت ای Const Pointer ہے ↗

Const Pointer وغیرہ unreachable data سے نجات

ک) Const Const وغیرہ Pointer ای خلیہ
ب) ای محدود حافظہ دیندہ

```
datatype * Const ptr = &x;  
ptr محدود پر ایسا کوئی نہیں  
readonly محدود عزیزی
```

Const datatype * ptr = &x;
کوئی نہیں ایسا پر ایسا کوئی نہیں

محدود کوئی ایسا

Ex int x = 10;

int * Const ptr = &x;

*ptr = 20; ✓

ptr = &y; X

Const int * ptr = &x;

*ptr = 30;

ptr = &y; X محدود ایسا

Pointers ایسا کوئی نہیں

کوئی نہیں

x = 30; ✓

النها

Const int x = 10;

x = 20; → invalid Const $\rightarrow x \sim 8$

int * ptr = &x; → valid کی جسے سمجھا جائے

*ptr = 30; → invalid Readonly

Note

$\& \text{ptr}[0]$ $\xrightarrow{\text{هو}} \& \text{arr}[0]$

stack نے خرچہ Data ایں ↪

int arr[5]; → static allocated array

arr کو size کا مجموعہ

Compile کی وجہ سے فیلر Data ایں ↪
heap ایں ↪

new keyword نے heap مامنونہ ایں

برمیج کو ایسا ممکن نہ رہتے فی المجن او null لو مانیں

EX new int[n]

malloc لئے گئے ایں ↪ array درجہ memory کو مانیں

array کو ورنہ memory کو مانیں

Pointer استعمال کیا جائے new والیں ↪

int * ptr = new int[5]; ↪ جارفا (ماکرو، array)

Stack

heap

arr

array کی رسمیت

index ایسا رہے

ptr[idx] ← ptr 80

graph

deallocation allocation معاكِر فتحة يفتحها stack ای او هو المسؤول بجزئي

deallocate \rightarrow الک کر کے میں جو اسی کا جگہ رہا اسی کو Heap میں \leftarrow
delete ptr \leftarrow لیکے جو اسی کا جگہ رہا
delete [] ptr \leftarrow array \rightarrow

میں اس کا ایک مثال ہے جو اس کا ایک سب سے بڑا مسئلہ ہے۔ اس کا نام "Memory leak" ہے۔ اس کا معنی یہ ہے کہ میں اس کا ایک بڑا حصہ خالی کر دیتا ہوں اور اس کا اسی حصہ کو اس کے بعد اس کا دوسرے کوئی دفعہ نہیں دیتا۔ اس کا ایک ایسا مثال ہے جو اس کا ایک سب سے بڑا مسئلہ ہے۔ اس کا نام "Memory leak" ہے۔ اس کا معنی ہے کہ میں اس کا ایک بڑا حصہ خالی کر دیتا ہوں اور اس کا اسی حصہ کو اس کے بعد اس کا دوسرے کوئی دفعہ نہیں دیتا۔

• الحالات الاجزئي garbage collector يرشح الاماكن تلقائياً
لوهست لنك لها

$\rightarrow \text{ptr} = \text{Null};$ Valid

char * P = new int [5];

لوجه Ram في الـ Variables كل ما يجيء terminate في Program will

undetermined behaviour \rightarrow delete[] ptr; \rightarrow ptr is null بیکوئد میتوان لاد 05 اور 18 ماکر لیکن اگر نہیں اس کا نتیجہ کوئی garbage data یا crash ہے۔

Dynamic allocated ~~by~~^{with} Functions ~~in~~^{to} Array ~~is~~^{ارجع} الـ ~~is~~^{ارجع} Caller's ~~is~~^{ارجع} Function's ~~is~~^{ارجع} Memory

Ex

```
int * NewArray (int n) {  
    return new int[n];
```

ما هو المترجمCompiler له نفس مختلف في المذاقات

int n; Standard C++ 11

$$C_{1n} \gg n,$$

int arr[

int arr[n];

Standard C++ 11

Compiler

لیخڑھاٹ اور stack عادی

150,000 Computers

```
new int[5]
```

call $\text{js}(\lambda x. \text{J}x)$ Function J will evaluate all variables in js .

إذا قيل ما يسمى لفظا حتى لو كان الـ variable امر سطرك ودخلها

کا تینہ

→ Pointer to Struct

```
struct emp{  
    int id;  
    stack >> id >> char name[20];  
};
```

```
struct emp{  
    int id;  
    char* name;  
};  
emp e1;
```

emp e₁ = { lo, "only" }

empty p = &e; ✓

`Cout << (*{P}).id;`

Cout << P.id; X priority w/ dot II
operator

This address is no

of element

* ~ (s)

$(*\text{ptr}).\text{id}$ $\xrightarrow{\text{خطهار}} \text{ptr} \rightarrow \text{id}$

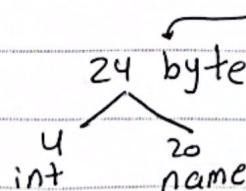
address also \leftarrow new arrow operator

$e1 \rightarrow id; X \rightarrow$ address ~~also~~ arrow $\Downarrow \alpha$

$(\&e1) \rightarrow id$; ✓

heap ॥ \rightarrow struct Josl \downarrow \rightarrow

`emp* el = new emp();` in heap



`emp* ptr = new emp[20];` array of struct in heap

`ptr[0].id` \rightarrow `ptr->id` \rightarrow element 1st

جواب اول عبارت عن

`ptr[1].id` \rightarrow `(ptr+1)->id`

datatype کے default value heap میں موجود نہیں کیا جاتا۔

`int` \rightarrow `char` \rightarrow `string` \rightarrow "10"

\rightarrow Pointer to Pointer

نایابی کو اسی Pointer کو کہا جاتا ہے

Ex `int ** ptr;` بیٹھا اور `ptr` 1st Pointer کو

`int * p` 2nd Data کو

`Cout << ptr;`

بیٹھا `ptr` نے address کی

`Cout << *ptr;` بیٹھا الک بیٹھا .. کی

`Cout << **ptr;` بیٹھا SVI Pointer کی SVI Content کی

Value کو بیٹھا کرے

heap کی dynamic ہے 2D Array کے لئے

Pointer to Pointer پڑھیں

`int *** p = new int*[row]`

Ex `int *** p;` Pointer to Pointer to Pointer to int

Good

APPendix

`Const Datatype *ptr = &variable;`

Pointer Point to Constant

بیت اور مسیر کی محتوی الگ سیکھو، علیہ Pointer

سے (ویر اچلینہ سیکھو، علیہ) حاصلہ ناتائج

`Datatype *Const *ptr = &Name;`

الگ سیکھو، علیہ مسیر کی محتوی (احلینہ سیکھو)

عوامیہ ناتائج

`Const Datatype Const *ptr = &Name;`

ویر اچلینہ محتوی الگ سیکھو، علیہ وہ (احلینہ سیکھو، علیہ) حاصلہ ناتائج

مسنونہ Function سے خلما مسیر کی محتوی

Class اور Variables میں قتوں اور داخل اور

Ex Class Demo{

`int x=10;`

`void Display() const { objectMember`

`x++; }`

X Error

3,

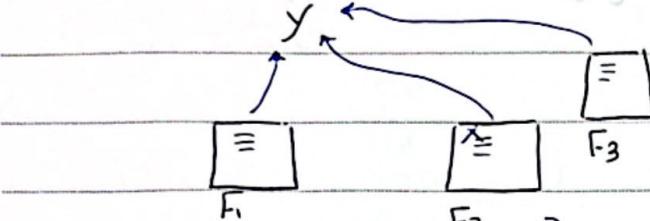
النحو

object oriented programming

Day 1

Linear Programming → Structured

to avoid repetition code in linear diagram



لقد رأى معاملة F_3, F_2, F_1

ي

لقد رأى معاملة

F_2 في F_1

error

لوحة P_2 في F_2 لفتح

فتح P_2 في F_2

معاملة F_2 في F_1

في لا هيئ

Maintainance

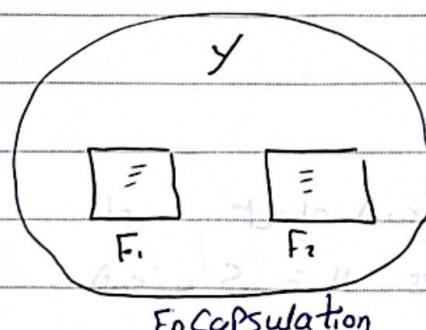
في كل معاملة

عابن (عمل) F_2, F_1 في y قد تغير، يتعامل مع y restriction

وأي دخول معاملة y يتعامل مع F_2, F_1 بطريقة

Encapsulation (كمبراسيون) OOP

يتم دخول دخل معاملة y إلى تصر معاملة y



حلت مسائل او

Data او Encapsulate بارها

مع او Function

8.00P

لدى Maintenance

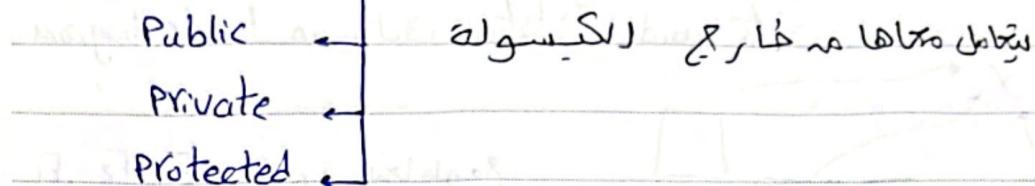
هيئ F_2, F_1 في F_2

* Pillars of OOP

II Encapsulation

فقط الـ **data** و **function** هما يحيطان بـ **object** .

يحيطان بهما **Access Modifiers** ، **hide** **data** **بـ** **private** .

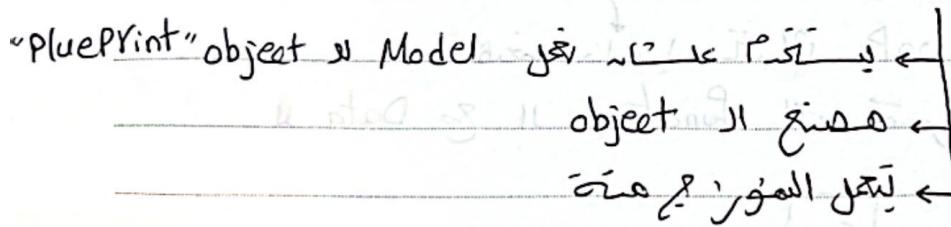


So **object** has **behaviour** & **Attributes** .

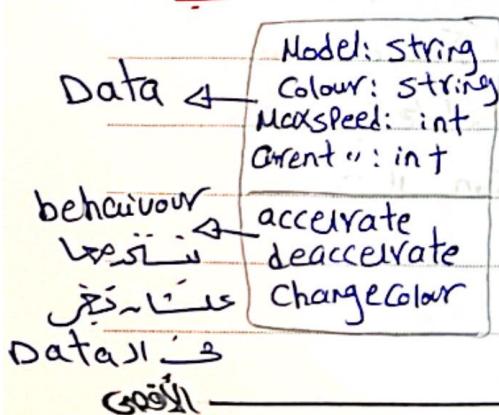
real life \rightarrow Map \rightarrow object

object has behaviours & Attributes .

class هو الـ



Ex Car



Car is a kind of object .
behaviours & Attributes of this object .

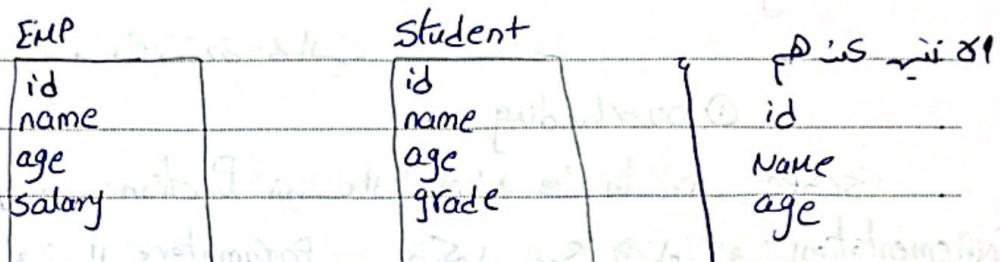
Class → Blue Print Memory یا لاجنڈ

object → instance in Memory

instantiated object \Rightarrow \downarrow is, \downarrow is object

Class 31 ~ Memory

② Inheritance

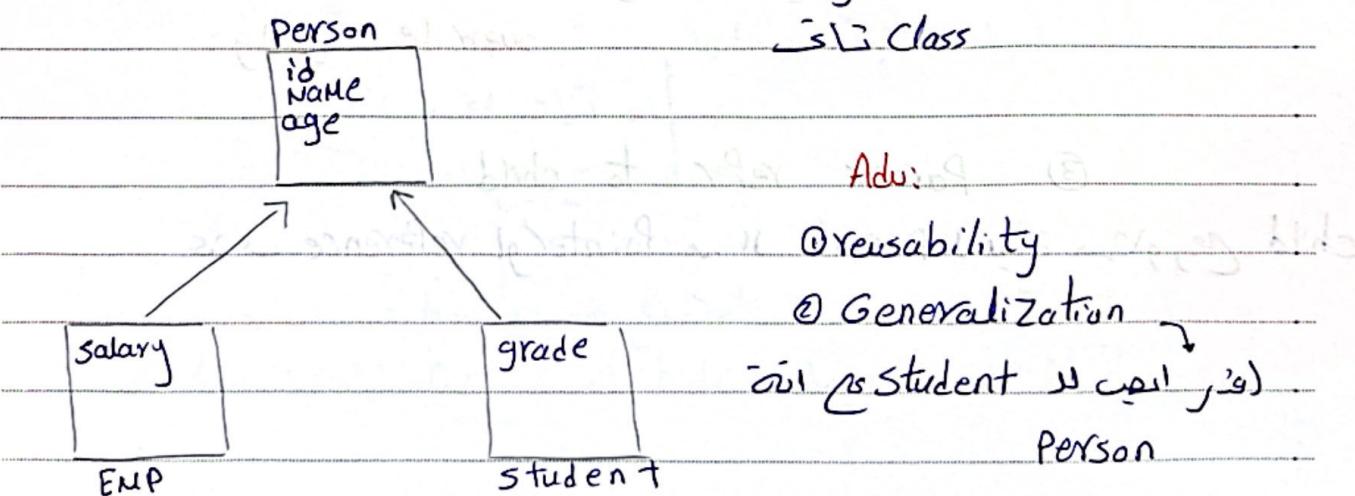


Constraints in `String`, `File` Code 3

مع اکیڈمی کردار اے تکمیل کوڈ اے اکیڈمی

کیا اس کیتی دا اس تھام اد ایتھر (multiple inheritance) اور

۱۹) استخراجات ایک موجودہ خرچ



Person → emp بیان ورثی میتواند سوچ Person
" → Student student ..."

instance ای پروتکل Parent یا سوچ ای object ای وارث میباشد

③ Polymorphism

نحو ای سکار

لتیفحة - کافین ←

① overloading

scope ای نفس ای نفس Function زکر نہیں

Implementation و کام سوچ مختلف ای Parameters سب مختلف ای درستیم درهم نوع

② overriding

ای تکرار وارث Function

New implementation

Implementation ای قدر ای Class

inheritance ای داد علیها میکند ای Function

override

③ Parent refer to child

child ای Parent ای no Pointer gl reference میکند

گفتگو

Solid \rightarrow ای محقق \rightarrow inheritance \rightarrow (قرآن)

\downarrow open for extension closed for modification

4) Abstraction

دستا کی صورت نہ

① رہنمہ ای اکا جات الماح و احمدہ

"Attributes" اکا جات الک متن

② (جمل) object \rightarrow details \rightarrow hide الک بیسائی

Ex رہنمہ car متن میں لعرف

فدا حلقیاً یتھلیل وہا ای ریھنگ لعرف اعز الرویہ

و بالآخر ارای و کیا "کفیہ ای ختم"

\rightarrow Hide Details Functionality from user \rightarrow developer

اے ارای اسکھا میھنیت فہ سیعیں ای میھو اھا

Ex Cout

دھمہ ارای بیان مھا ملوش علاقہ فہ منکو دئے میھو ارای

ارای بنادی علیہ اسکھا

نیا ای
بر صحیح ای

Function ای Data ای \rightarrow Encapsulation

Function ای Details \rightarrow hide جمل \rightarrow Abstraction

UML "Unified Model language"

Analysis → Design → Implementation

Class diagram چیزی کہ

ای ریپریزینٹ اے وہ اے class کی جسے

EMP

id: int; private
age: " "
name: Char[20]

عابز گھست لئے سیکھا
وہاں عن دا حل اے Class دا
اے Function ہے

وہ Restrict ہے
Validation of Data

setId(id): Public
setAge(-Age): "
getId(): "
getAge(): "
setName(char[20]): "

لیے کیا جائے
Data or restriction
کیا validate ہے
غیرہ وغیرہ



F1 F2

setId() یو جو ملبوس ہے id کی لوگوں کے لئے F2, F1 کے لئے نہیں

Data hidden اے (خاتے) Encapsulation) (الیکسپوشن) کیا جائے

Access modifier
(حقیقتاً)

Function
کیا الائنا وار
وہ یو جو کہ

کوئی

class emp {

int id;

char name[20];

int age;

by default → private

Class or جمله default Access Modifier ۱۰۰

};

private جمله

class emp {

section جمله

Public :

↑ Public جمله

int age;

int id;

};

? Private جمله attributes as restriction conditions ساخته الرغب

Maintainance ۱۰۳ جمله

class emp {

int id;

char name[20];

int age;

Public:

Void SetId(int_id){

id = id;

جیت او حیثیت (Good)

لینڈا Data validation

سے

Naming Convention ↗

Attribute Name ایجنے

};

Note

length جملہ

strlen ↗

String.h

$e_2 = c_1$

e_1

بياحد value equality

e_2 يساوي e_1 في value أو في متغير

id: 3000

name: ahmed

age: 35

salary: 1000

void print(emp e)

y = ↓

emp سعر Parameter

e_2

id: 3000

name: ahmed

age: 35

salary: 1000

Const بمعنى Read only أو لخطبة attribute جس

Stand alone
function

المكتوب خارج class يقول على Function

لقد استعملها باسمها لوحة

member Function في Class داخل أو all Function

object داخلها يمكن أو

Pointer بمعنى automatic بمعنى hidden parameter

all Function في المكتوب أو المكتوب Caller object

"this" المكتوب في طرف Class داخل أو

class داخل object داخل Class بين مربع من متغير class

الآن

behaviors + attributes عکسی و خواص کلاس

مقدارها را برای این object بدهید

Note

Friend Function

private member

in access نحوه دسترسی

standalone function

خواهد داشت که object نداشته باشد

C++ → Not Fully oop

because

standalone function

① Can declare Function outside class

② Friend Function

Ex

class Test {

private:

int a;

protected:

int b;

public:

int c;

friend void Func();

};

good

void Func() {

Test t;

t.a = 10; ✓ valid

t.b = 15; ✓ ..

t.c = 16; ✓ ..

g

Day 2

→ this Pointer

this is Data Field or state of object

Code/text segment نصف متغيرات Methods اد

Data اندريها لغت يعرف هنن اه اناعانز اد Function اد

? e2 & g c1, d1

هذا في اه اد الـ object

hidden Parameter نصف Function اد بحث عن اد this

اسمه class يكتاو،

Function اد الـ object

e1	id: name: age: salary:	emp * this
e2	id: name: age: salary:	Print1(); g =

this اد لغت Function

لوحة memory

Pointer

this->age;

address
caller

hidden Parameter نصف member function اد موجود مع this

original caller object اد this change Pass by address

object اد Function اد تكرار كود اد optimization

this Parameter اد standalone Function

Void setId (int id) {
 id = id;
 this->id = id;}

→ garbage data هيچ Parameter = Parameter اه هنن اد

Compiler Conflict اد كل اد

اللغة

```
Void class emp {
```

```
    int id;
```

```
    void print() {
```

```
        int id = 10;
```

```
        cout << id;
```

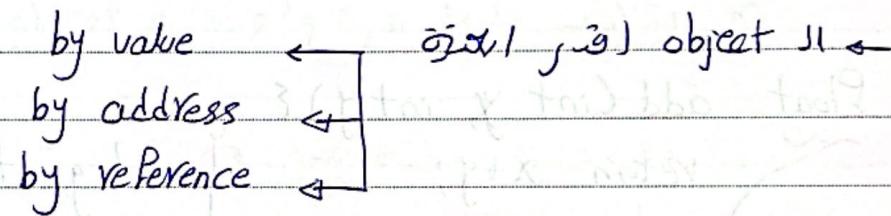
دالة هيكل
local variable
function

q:

على - (حيث نسخ الـ object)
this->id

oops! violate your standalone function

ستتحقق



⇒ overloading

Polymorphism هو اسلوب او

Signature نفس Function له نفس Signature

نوع return
نوع Parameter
نوع Name

و return type يختلف كغيره من عناصر الايك ينفرد

overloading دالة واحدة وتحتوي على عيارات مختلفة

المعنى

```
int add(int x, int y){  
    return x+y;  
}
```

```
int add(int x, int y, int z){  
    return x+y+z;  
}
```

```
main(){  
    add(5,4); } Function  
    add(5,4,6); } Parameter list جملہ  
}
```

Float add (int x, int y) {
 return x+y; } ambiguous error

overloading جو کسی دو فونکشن کے بین میں ہے اسے overloading کہا جاتا ہے

```
Float add (int x, Float y) {  
    return x+y; } valid parameters ترتیبی اور جو کسی دو فونکشن کے بین میں ہے اسے overloading کہا جاتا ہے
```

```
Float add (Float x, int y) {  
    return x+y; }
```

```
Float add (Float x, Float y) {  
    return x+y; } ambiguous جو کسی دو فونکشن کے بین میں ہے اسے overloading error کہا جاتا ہے  
    datatype کو جو کسی دو فونکشن کے بین میں ہے اسے overloading error کہا جاتا ہے  
    casting کو جو کسی دو فونکشن کے بین میں ہے اسے overloading error کہا جاتا ہے
```

Parameters نفس ال Name Function لوال Error بحث
 Implicit casting يغير Data type ال اندروج

عادی class داخل ال سُلول overloading ال

⇒ Constructor

creation ال لفظ object attribute initialize العمل على

SPECIAL Function عبارة عن

Void return type مهام ①

class ال نفس Function ال ②

object ال automatic بيتادي على

"creation" لـ "creation" لـ "creation"

creation على object ال هررة واحدة الوارد

Function ال object Create سُلول (Constructor)

لتفت فـ دایرۃ حیاة ال object لـ "life time"

فـ دایرۃ حیاة

لـ میادی private ال سُلول Constructor ال

class ال object لـ دایرۃ حیاة design pattern

class A {

A(...){

y =

y;

object member Constructor ال

this ← hidden parameters

initialize (plz) بـ Data ال

garbage data ~ Constructor ال

conclusion

Constructor میں زکر دے سے Overloading (فرار اسکا) Constructor ہے

Class Name {

Name (---) {

Parameter میں کسی Constructor کا

to create object

Parameterless constructor اسے

Name n (---);

to create object

Name n;

creation ایسے attributes کا دیتے ایسے setters ہے

objects life time کے لئے مرغہ والے میں استادی کے Constructor ہے

ایسا کوئی Function کے ایجاد کے Constructor میں داخل ہے

Constraints کوئی نہیں تو setter کے Class کے

Code کے لئے سہی

لیس لس (LIS LIS) Constructor ایسے ہے

default Parameterless Constructor

میں کوئی کامیابی

کو ایسے default Constructor کے لئے ہے Constructor کو ایسا

عکس (also) پر Parameterless

گزینہ

→ default Argument

Mandatory Args (جواب مطلوب)
فوجی پریوریتی

```
Void add (int x, int y, int z=0) {
```

```
    return x+y+z;
```

y

optional is
جواب مطلوب نہیں

x y z=0
add (10, 20); 30

add (10, 20, 30); 60

لهمہ جو شے دھیر

گذرا کریں جو کہ مجب میں default values ہیں
default values

```
Void add (int x=0, int y) {
```

```
    return x+y;
```

y

Error

Default value پریوریتی

ex int y=0

```
Void add (int a, int b=10, int c=20) {
```

```
    return a+b+c;
```

y

Default Value کو بے پریوریتی کریں

اکسیں سوال کرنے کو کیوں کوئی کوئی

overload ہو تو function implement کو

overloading اور default value ہی پریوریتی default value کوئی

کوئی

Function و اپریو اسی Special Function اور کنٹراکٹر اور Default value اور پری جسے سوچو۔

Class Name {
 Name (int real) {
 y =
 Error
 Creation (اندیشیدن)
 real object لوحهٔ
 و فقط هست همیغ عارف
 نیادی می‌گیرد
 Name (10); } } ambiguous error
 Name (10, 20);

⇒ Destructor

ممت هو الک ییزدیں ال object ہو یعنی فیل کھاتا لئا لنسح ال object

memory میں object کا اسی special function ہے

لها نفس اکواریوم Class و فلائم

~ Name ← Class ۱۱ فیض ①

① پیامدات Parameter و میانجی حاصل

overloading مفہوم ۲

③ پیشادی انتبار اراله از object در memory

٦٣ يشادى هرة واحدة فقط في اية من

is this the model of data objects الکیڈز مرے ہیں

الْأَفْوَى

```
class Name {
    ~Name() {
        ≡
        ↴
    }
}
```

int myFun(Complex c) { → object ↗ create جملہ
 ≡ constructor کو سمجھا جائے
 ↴ ملنا ہے کہ Copy کیا جائے
 ↴ destruct گالئی پڑا ہے

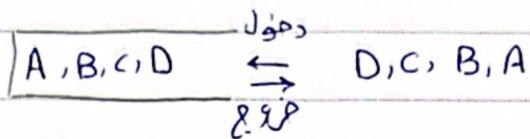
int myFun(Complex & c) { → create ↗ two reference جملہ
 ≡ already ↗ object کو موجود ہے
 ↴ Function کی destructor کیا جائے اور Constructor کیا جائے

↙ deallocate یا (اسیکو destructor کی پریس
 ↴ File gl Connection کو close gl memory کیا جائے

Good

→ Stack DS

"LIFO"



APP back undo button → stack

undo جس لیگ stack دھن حاصل کرنا ہے جسے خلف

بے دفع لامن حاصل کرنا ہے

Calling Function

کوئی Function کا address پیدا کرنا ہے

کوئی دعویٰ کرنے کے لئے Calling کا لگانا ہے

→ tos "top of stack"

Data کا جگہ دھلتے ہیں (لیکن اس کا مکان)

tos++;

Push() → to add element in end

Pop() → to get last .. stack

tos--;

Push کا عمل stack کا سائز tos = size - 1 کے لئے

لو ہے تو دو تا ڈیزیل کے لئے

stack

attribute	size: tos: arr:
method	Push() Pop()

deallocate، الگو، destructor کا

heap کا ذخیرہ، dynamic location کا

graph

```

Void myfun( Stack s) {
    int x=0;
    while( s.pop(x) != -1) {

```

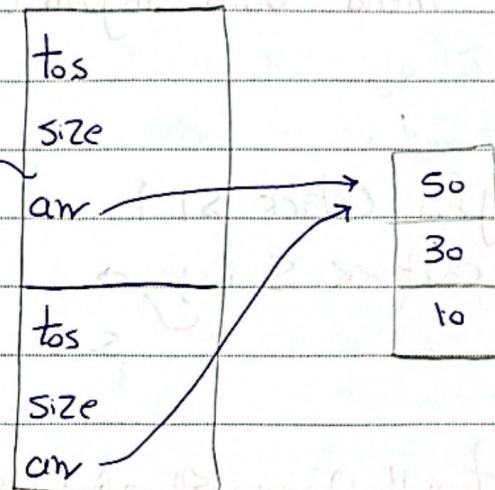
Cout << x;

دالو دايلر هسيم دايلر
Pointer دايلر هسيم دايلر
لو reference
original two alias.

```

main() {
    stack s; myfun
    int x;
    s.push(50);
}

```



• يتطلب ميكلاج لويادت لو ياديتس
وهو خارج من الـ Value Function
يتخاور مع original data او الموارد

(Bitwise Copy) shallow copy

لو ياديتس address Copy بيعبر بلوك
تحت الموارد نفسها

Deep Copy Copy Constructor or Code

الآن

لو عابن رخکے اور Standalone Function اور Friend Function
اے رخکھا

class — {

private:

int i;

public:

Concept of OOP اور violate

Friend Void myFun (Stack s); ↑

};

Void myFun (Stack s) {

Cout << s.i; → this میں ہے سے
}; → s.id یوں id لکھتے ہیں

Friend Function اور Class میں لکھتے ہیں
وہی Friend

OOP Pillars کا Friend Function اور اسے بینج لے

Class میں Friend Function کا implementation

inline / outline Function کا مارچ اور اسے نہ کھا کر

this کا نہ کھا کر class میں دوسرے کوئی دفعہ

implementation اور header file کا مارچ اور اسے

scope resolution کا مارچ اور اسے

Function Pointer to Function کا مارچ اور اسے

سے نہ کھا کر class میں دوسرے کوئی دفعہ

اللئے

`int * p = new int[3];` array in heap 3 element

`int * p = new int(5);` → object initializer



س زیگ بزیg → یک سیج همچو

Constructor

`Complex * ptr = new Complex();`

اسی address جیg heap جیs object کی خوبیت

ptr جیs مخطوط

stack جیs ارجع نه object جیs ارجع

object جیs زیگ ptr جیs جیg

`ptr.print();` x

`ptr->print();` ✓

ذکر دادی

`delete ptr;`

کس ایک بیو، علیہ فوجینا دی ای

destructor

→ array of Complex

`Complex * ptr = new Complex[3];`

Parameter less constructor

Day 3

1 Recursion

لما زادت العدد Function

$$\text{Factorial}(4) = \text{Factorial}(3) \times 4 = 24$$

$$= \text{Factorial}(2) \times 3 = 6$$

$$= \text{Factorial}(1) \times 2 = 2$$

$$\text{if } x > 1 \rightarrow P(x) = x \times P(x-1)$$

$$\text{if } x = 1 \rightarrow P(x) = 1 \quad \text{"base Case"}$$

جاء في المثال العدد 1 هو الحالة الابتدائية

وهو يرجع return 1

Recursion (ما زاد العدد فالحالة الابتدائية Function)

```
int Fact(int x) {
```

```
    if (x == 1)
```

```
        return 1;
```

```
    return Fact(x-1) * x;
```

4

Fact(4)

x=1	return 1	Fact(1)
x=2	Fact(1) * 2	Fact(2)
x=3	Fact(2) * 3	Fact(3)
x=4	Fact(3) * 4	Fact(4)
		main

→ if No base case → stack overflow

→ also recursion no base case will lead to stack overflow

conclusion

Applications

→ Divide & Conquer

Ex merge sort

quick sort

Ex tree DS binary search Dynamic Programming

Ex Power(x,y)

$y=1 \rightarrow x$ "base case"

$y>1 \rightarrow x * \text{pow}(x,y-1)$

recursion \rightarrow يعنى iterative algorithm \leftarrow

use less space \rightarrow less stack

Save state \rightarrow يعنى recursion \rightarrow

return address \rightarrow

or Performance recursion \rightarrow

iterative \rightarrow

time complexity \rightarrow نفس اى recursion \rightarrow less iterative \rightarrow

less stack \rightarrow

in program \rightarrow غير ملحوظ \rightarrow stack \rightarrow نفس اى recursion \rightarrow Note

15 configurations \rightarrow سنتو

الآن

2 Copy Constructor

Function ۾ اس کو pass by value ہے object یعنی dynamic allocated ہو بینا دی اور بیند فر اور destructor ہو بینا دی اسکے لئے access کا رجی ہے

الوائٹ کاٹ کو دھنھا
destructor ہے

```
myFun(stack s) {  
    int x=0;  
    while (s.pop() == 1) {  
        y =
```

y

main() {

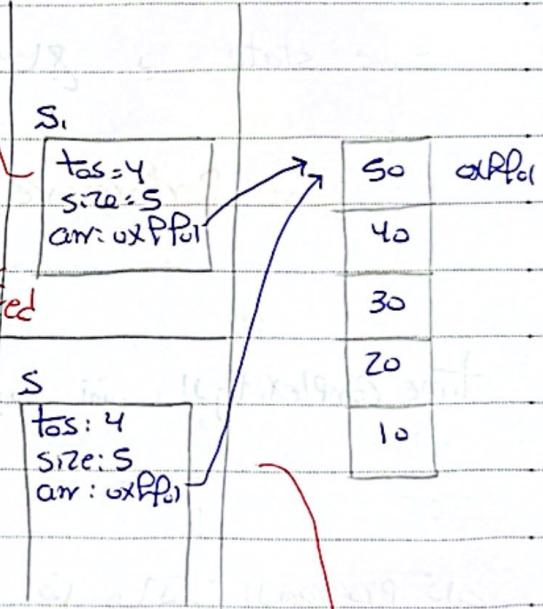
Stack s; یہ سکٹ کو دھنھا
s.push(50); destructor ہے
s.push(40); کو دھنھا کرنے کا
" 30; dynamic allocated ہے

" 20;

" 10;

myFun(s);

s.push(60); → Error



y deallocated یعنی array کو الی جمع کرنے کا error یعنی free ہے

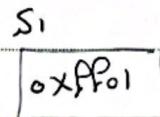
Copy Constructor ہے copy کا عمل ہے

الآخر

Shallow Copy s_1 by default هو مموجوـد Copy constructor لـ Pointer المـؤـرـي الـكـفـيـة يـسـعـوـا بـ Copy بـ نفس الـقـيـمـة dynamic original data location.

Void myFun (Stack* s1) {

\equiv
عـاـيـزـهـاـعـلـىـكـفـيـةـ دـالـكـ سـتـارـ

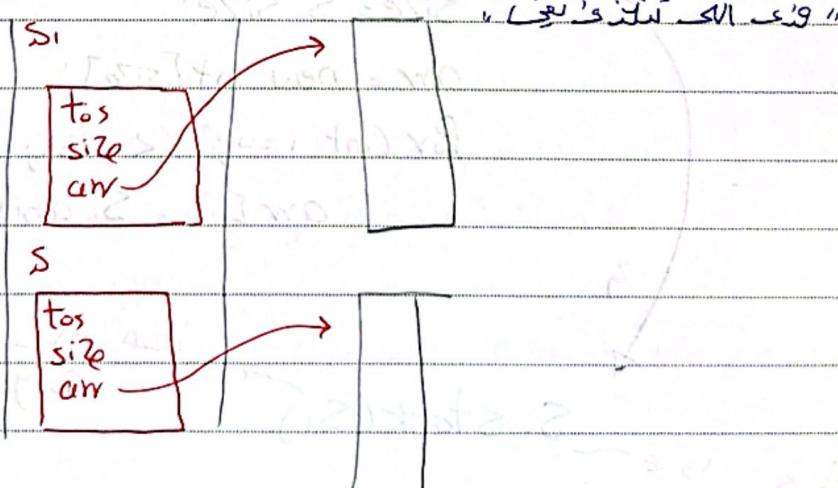


deep copy عـاـيـزـهـاـعـلـىـكـفـيـةـ Copy Constructor او سـتـارـ فـاـحـمـاـ هـكـيـةـ تـقـيـسـتـا

dynamic allocation عـلـىـكـ لـوـغـيـةـ

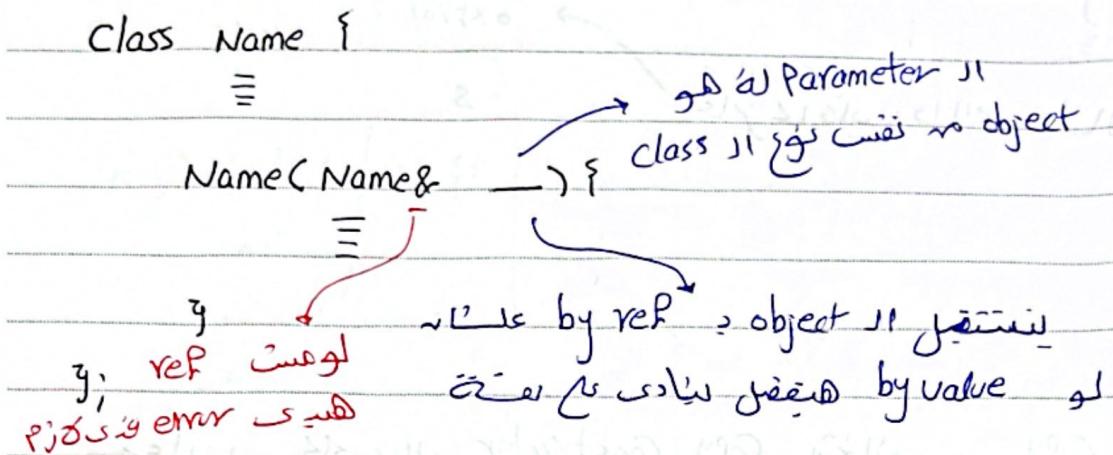
يـادـنـ مـارـ قـيـمـاـ

عـلـىـكـ مـارـ مـاـ مـنـ



الـفـيـلـ

values نہ Copy کیا جائے (Copy Constructor) اور
 isis Function کو اس Class کی object کے
 object کو object by value کے طبقے میں دیا جائے۔
 لیکن Copy Constructor کی تفاصیل



if stack

```
stack(stack& s) {
    tos = s.tos;
    size = s.size;
    arr = new int[size];
    for (int i=0; i<=tos; i++)
        arr[i] = s.arr[i];
```

this \rightarrow

s.stack(s1)

COPY

this سوارہ Private اپریٹر کو اس Class کا داخل حصہ اور class داخل کیا جائے۔
 اسی طبقے میں object اسی طبقے میں دیا جائے۔

الآن

Class Name {

\equiv

Name (Name & S) {

\equiv

Const Names S \rightarrow لو خلیخه

هست هر را عرض کردار

S در اسک

y;

myFun (Name st) {

\equiv

کارها

y

Name st = new Name (S);

Main () {

دالک یعنی

Name S;

ویل س جو میگیرد

myFun (S);

درست هست
st کو میگیرد
ویل س

y

stack S2 (S1);

or stack S2 = S1;

هستادو از
Copy Constructor

S2 = S1

Copy Constructors هست هست
assigning دادن
object creation (ایجاد یک چیز)

شallow copy یعنی equality operator (==)

cool

یہاں کو Copy Constructor کا مفہوم

object by value کے لئے ①

object ~ object declaration کا ②

Name S = S, OR Name S(S)

object by value کو return کرنا (عمل ③)

Name myFunc()

return by ← ≡

value return new Name();

لوگوں کے نام alias Name اور Function کو reference کرنے والے

کارکردگی کا object ہے اس کا نام copy constructor ہے

int x;

int y;

x=10;

int &r = x;

y=30

OS table
X: 0x FF00
y: 0x dd11
r: 0x FF00

لوگوں کا
لیتھیل دفعہ

only one copy constructor کا

heap میں ہے اس کا نام copy constructor ہے

default copy heap کو لے کر deep copy کرنا عمل کا

Note

Name(Name& n) کو فائدہ حاصل کرنے کے لئے
y } garbage data

النقد

3) Class Relationship

"object" "

class A { } OR

class B { }

A a;

} }

B \rightarrow object

A \rightarrow object \rightarrow B \rightarrow object

"Composition"

class B { }

A * a;

} }

\rightarrow B \rightarrow object

A \rightarrow object

① Composition relationship

② Aggregation ..

③ Association ..

④ inheritance ..

II Composition

object \rightarrow object

Construction \rightarrow creates two objects || and \rightarrow destruction \rightarrow

destruction \rightarrow

one object \rightarrow one object

" " " " " " " " " " " " "

Ex

House = Room = Room \rightarrow Walls

Room \rightarrow walls

Goal

وادی دایرکشن احادیه من ایجاد کنیم \Rightarrow Composition یا

team گلوب Match یا Ex

دو تیم کیونکو مودودی two team ایجاد کنیم ایجاد کنیم \Rightarrow

Match بینیت میکنیم team کو

دو تیم کیونکو مودودی two team ایجاد کنیم Match \Rightarrow

3) Aggregation

کل لستیل انکل کلیزے اکنون ہو مودودی

"کل و جن" کیا معنیت (Aggregation) یا Composition یا

کل و جن کیا

Ex Stack, array \Rightarrow Composition

4) Association

Student - Instructor

لوازنی هست موئیض سوادی Construction یا

سین وقت ما ایتنی هستیا ملو مع ریب

بی دیکشن (قسر افریزہ) ایک جنیہ

aggregation یا implementation یا

انکل کی ایجاد کنیم ایجاد کنیم \Rightarrow Composition یا

انکل کی ایجاد کنیم \Rightarrow destructor

انکل کی ایجاد کنیم \Rightarrow create یا new یا create یا new \Rightarrow Aggregation یا

انکل ایجاد کنیم \Rightarrow ایجاد کنیم ایجاد کنیم ایجاد کنیم

انکل کی ایجاد کنیم \Rightarrow ایجاد کنیم ایجاد کنیم

Class Rectangle {

Point ul;

Jgl \rightarrow Constructor

Point LR;

Rectangle(int x1, int y1, int x2, int y2) : lr(x1, x2),
ul(x1, y1)

y;

y

Constructor chain ↗

setul(Point * p){

ul = *p; \rightarrow value equality

y "copy values"

\rightarrow trick

returntype Name();

default: int j1 ↗

default void

return \sim

\sim copy j1 ↗

sole C++ j1 ↗

Class Rect {

Point * r;

main() {

Public:

Rect();

aggregation ↗

Rect() {
cout << "Ctor";
y

r.setPoint(8, r);

setPoint(Point * r){

P.setX(100);

this->r=r;

r.print();

graph LR
y1 --- y2

x(10, 20)

للح design اور الج relationship

Note block code

{

Ex: {

三

Rectangle r;

y

y

لایو جس کیا سیر اور

لایو جس

الآن

Day 4

→ inheritance relationship

Student class

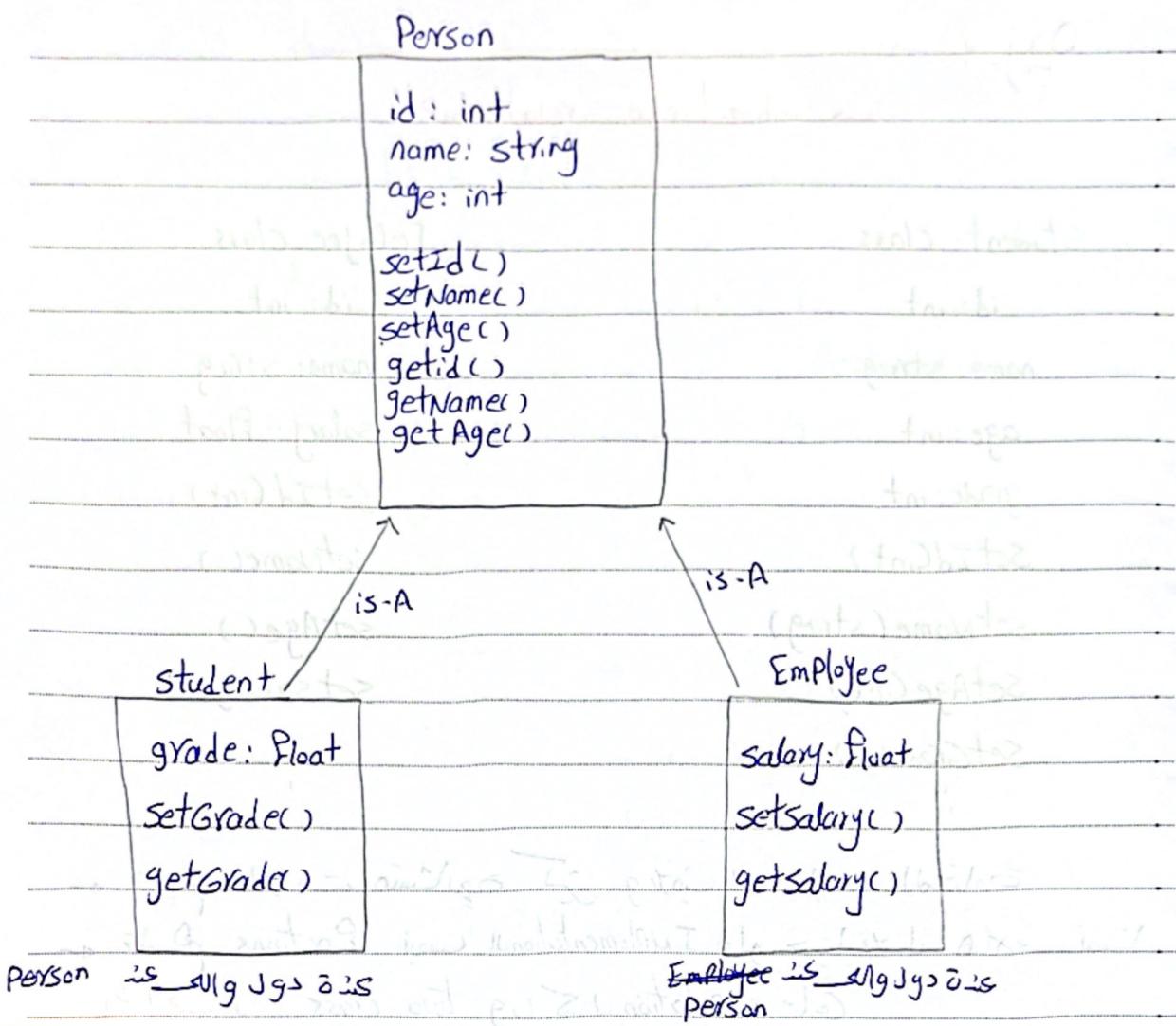
id: int
 name: string
 age: int
 grade: int
 setId(int)
 setName(string)
 setAge(int)
 setGrade()

Employee class

id: int
 name: string
 salary: float
 setId(int)
 setName()
 setAge()
 setSalary()

هي هي حمايات = هستاريه كن ونف احاجيات المختلقة ←
 هي هي Implementation Functions نفس اليفس اليف ←
 Code repetition ↗ در دو class ←
 لاق استغ ← او inheritance ← او ←
 فون احاجيات المشتركة و التأسيس Class جاري ←
 بورنو هندي

Good



Person → Parent class [base/super] class

student/Employee → Child . [derived/sub] "

Adv

- ① reusability ↗
- ② maintenance ↗

cool!

11

ـ مـسـتـقـلـةـ اـرـوـعـ لـعـمـلـ حـاجـاتـ مـسـتـكـلـةـ اـرـوـعـ لـعـمـلـ .
ـ ظـرـفـ تـكـوـنـ لـهـ مـاـتـ "A-is"ـ فـإـنـ الـوـاقـعـ .

Dog class

Name _____

getNames

Dog is-a person X

Person

مقدمة المقالة

مکانیزم ارثیت اور ارث علیہ class میراث (Inheritance) کا نام ہے۔

"collection" obj File is class like this

و ممکن است این کار را با هدف تقویت امنیت ملی انجام داد

لینید و اسٹیف فٹچر ار نوائیکس Features

→ Syntax inheritance

Class A's

三

21

Access Modifiers

پہنچ اکاہات الکھو رخا

1878-1881

جی ٹیکنالوجیز

class B:

三

Y; 2025

20,9

private public protected

www.jstor.org

—

in class at

مکانیزیم اے حاصل فی A نہیں موجود فی B لیکن اے مکانیزیم اے حاصل فی B نہیں موجود فی A

وہ اس Private rental میں مکانات کا انتظام کرتے ہیں جو اپنے عادی سب سے مقدمہ رکھتے ہیں۔

↳ If child class give no object as argument member

→ 2 we child class 11

Class A {

int id;

protected:

int c;

y;

Class B: Publ.

Public:

هفترست او هلهه هارج

Child \rightarrow Parent \rightarrow

private static void

ماهیت Public سیاست
ماهیت Private از

protected → private → Artacts (≡ 1 - 151: A)

Private 15% Pooled 2015

دُوْنَ الْوَرَّةِ دَاعِيًّا void print()

ای جاہنگیر Public id = 4

Concepts of Cryptography

97

Private

نی اکریوئی میکا جو ماجد / Parent

```
main() {
```

A B b;

b.id = 10; x private

b.c = so; α protected

3

الـ child هو الـ لـ قـ يـ وـ مـ

للماء دے میں جو دیوار و سطح کاہ مکاہ

مقتنيات المكتبة

\Rightarrow Constructor

"relationship Parent & child"

کنکٹر کا تصورت مدت یہ رہا ہے لیس دفتر Constructor

Calling 1st class

Parent object میں کوئی child object نہیں ہے اور child object میں کوئی parent object نہیں ہے۔

Default constructor اور میڈیم constructor by default میں دوسرے دعائیں دیا جاتا ہے۔

لوگوں میں موجود ہر چیز کا مکان
واحد ہنادسی Chaining
Constructor اپنے

Parent ہے اس کے

Parent ہے اس کے child میں قیمتیں۔

class A {

=

A(---, ---) {

y =

y;

class B : public A {

B(---) : A(---) {

Constructor Chaining

→ class بے کار

ar C++ میں اسے

multiple inheritance

لوگوں میں دوسرے دعائیں
child کا میں دعائیں دیا جاتا ہے۔

multiple inherit

(base, super) پر

لوگوں میں دعائیں دیا جاتا ہے

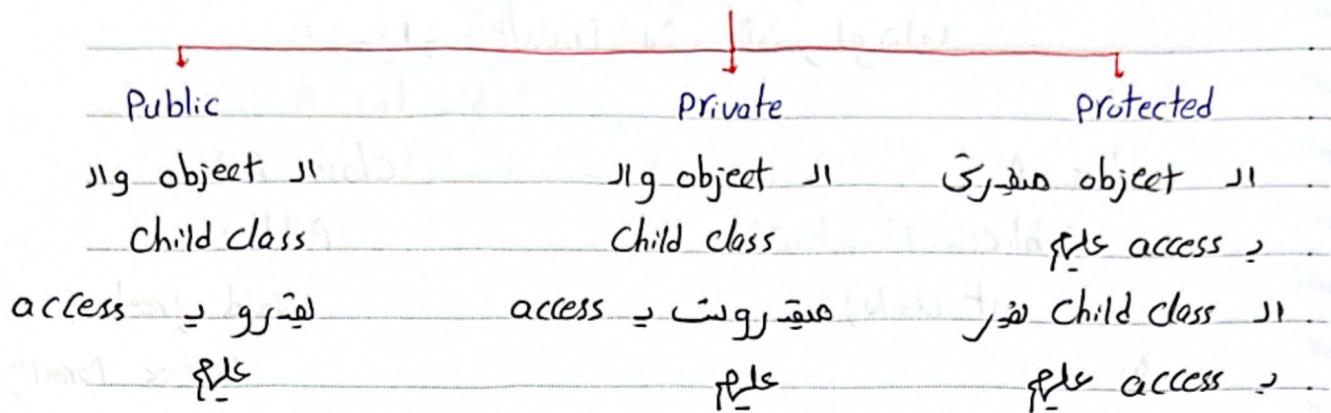
Parameter less constructor → automatic ہے

لوگوں میں Parent ہے اس کے

error ہے۔

⇒ Appendix

→ Access Modifiers "Specifiers"



→ object can't access Private & Protected member

Public → Can access by object

مکرریتی برداشتی Access Modifiers

Base class کیا ہے؟

class Base {

private:

int a;

protected:

int b;

public:

int c;

Void FuncBase() {

a = 10;

b = 20;

c = 30;

class Derived: Base {

public:

Void FuncDer() {

a = 1; X

b = 2; V

c = 3; V

3

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

3,

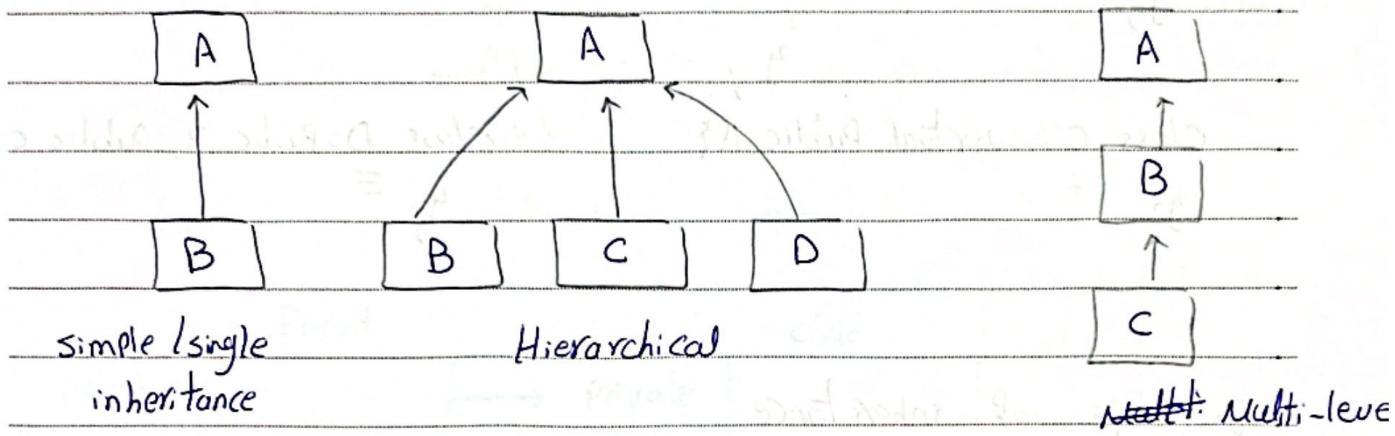
3,

3,

3,

	Private	Protected	Public
Base class	✓	✓	✓
Derived ..	✗	✓	✓
object	✗	✗	✓

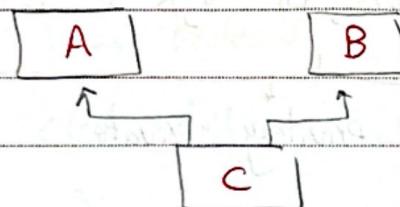
⇒ types of inheritance



simple /single
inheritance

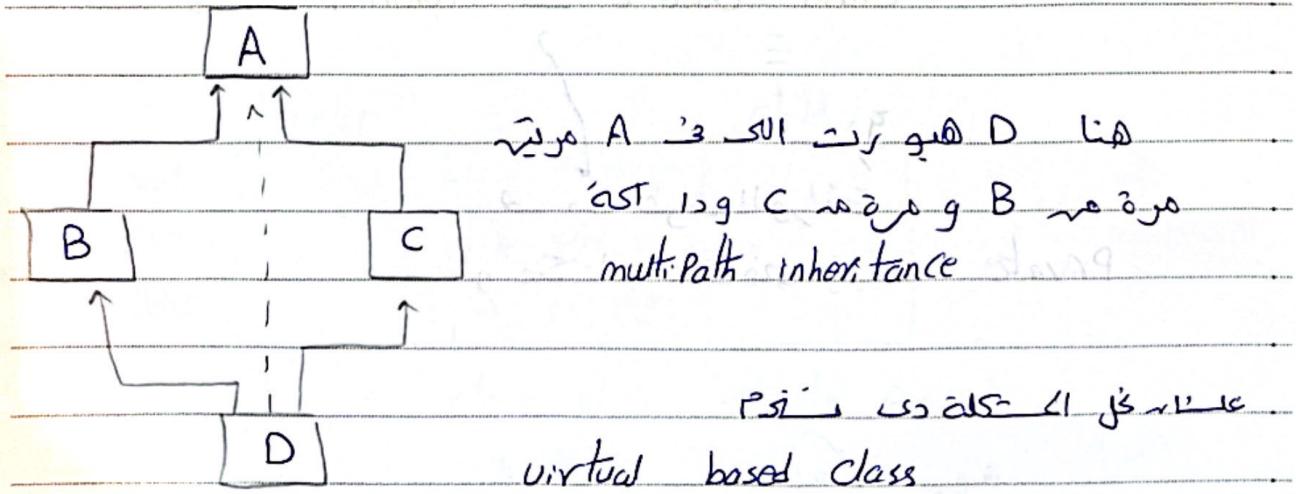
Hierarchical

~~Note:~~ Multi-level



multiple inheritance

Java → allows



Goal

1

→ virtual based class

→ useful for removing the ambiguity of the features of parent class in the right class, that is coming via multipath

Class A

$$y; \quad =$$

class B: virtual public A {

一一

Class C: virtual Public A{

$$y_1 =$$

class D:public B, public C;

$$y_j \equiv$$

\Rightarrow ways of inheritance

عِنْتَ ۝ حُرْفُ الْوَرَأَةِ

Publicity "Public"

privately "private"

protective "protected"

Class child: _____ Parent?

11

51

دی ھرقت الورانہ

لو مکناتس حامدہ یعنی Private

	Parent		child
private		→ private	
protected		→ protected	
Public		→ public	

Class child : Public Parent ↴

یہ ≡ }

فلاسی کوئلے کے ساتھ

Protected اور protected ↴ ↴

Public .. Public ↴ ↴

	Parent		child
private		→ private	
protected		→ protected	
Public		→ protected	

Class child : protected Parent ↴

یہ ≡ }

protected ← protected ↴ ↴ Public اور اس کے ساتھ

	Parent		child
private		→ private	
protected		→ private	
Public		→ private	

Class child : private Parent ↴

یہ ≡ کوئلے کے ساتھ

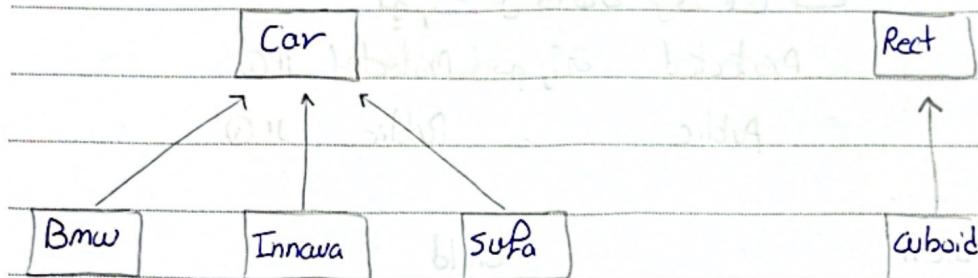
private لیکے

Good

ways of inheritance II

grand child child ای تواریخ ای سواد object ای سیتوختن +
نفر تکسٹ ای grand class ای سیتوختن +
child ای سواد child class ای سیتوختن +
child ای سواد child ای سواد

⇒ Generalization vs Specialization



→ bottom up

→ base class don't have any thing to give their child class, Just their purpose is to group them together

→ easy manage this things

→ child existing then we define a base class

→ Same Name, but different object actions or different things

→ achieve Polymorphism

→ top down

→ base class has something to give to child class

→ base existing then child we defined later

→ Share its

Features to its child

good

→ Generalization

process of extracting common characteristic from two or more classes and combine them into a generalized Superclass

→ Specialization

reverse process of Generalization means creating new subclasses from an existing class

→ Data Hidden

class Rectangle {

 Public:

 int l;

 int b;

 int area();

 return l * b;

 }

 int perimeter();

 return 2 * (l + b);

 }

};

access; ~ as Private ~ Data ~ خواص ~

property functions ~ ينطوي على getter & setter ~ البيانات ~

good

main() {

 Rectangle r;

 r.l = 10;

 r.b = -5;

 System.out.println(r.area());

 System.out.println(r.perimeter());

data hiding ~ خواص ~

وهي تمنع اـ (التعديل) ~

function ~ خوارزمية ~

Day 5

int $x = 20;$ // 4 byte

long $l = 60;$ // 8 byte

C++ ISV syntax

$x = l;$

error ممكناً

$l = x;$

loss ممكناً

هيلصل loss اور

4 byte to 8 byte

فقط هيئه هيلصل

Explicit

Implicit Casting

loss ممكناً

هيئه هيلصل

بوضع البيانات

البيانات

→ Casting

ممكن تغير في شكل البيانات

ويمكن loss لوحول

large size to

small

$x = (int) l;$

$l = (long) x;$

Explicit
Casting

loss ممكناً لوحولImplicit casting في C++

Ex float $f = 2.3;$

can write without
loss "Implicit"

int $x = 10;$

$f = x;$

$x = f;$

Can be loss data

Explicit حفظ

loss معروفة (هيئه عارض)

$x = (int) f;$

loss loss loss

النهاية

→ Another syntax

`newdatatype(var);`

وهي نفس الأداة

```
class shape {  
    =  
    g;
```

```
Class Rect: Public shape {  
    =  
    g;
```

`Rect rect(10, 20);`

shape is المترافق مع

so it is only

`class triangle: Public shape {
 =
 g;`

`void PrintArea (triangle t) {`

`cout << t.getArea();`

`Rect r(10, 20);`

`PrintArea(r);` → Error → specialization function

triangle لا يملك

Generalization & inheritance لا يملك

لأنه ينبع من Parent أو نوع Parent object

as its child

`void PrintArea(shape s) {`

`cout << s.getArea();` → if shape is parent
if getArea() is error

`PrintArea(r);` → shape is all kinds

" (+)" →

الآن

shape S;

dim: 10
dim2: 20

Rect $r_1(10, 20)$;

dim1: 10
dim2: 20
Z = 100

$$S = r; \quad \checkmark \text{ "valid"} \\ \text{error} \quad \leftarrow \\ S \neq 0 \\ \text{! ملاحظة} \\ S.z = 50; \\ \text{triangle } t(10, 20); \\ S = t; \\ \boxed{S = t};$$

S
dim1: 1
dim2: 1

جَسْتِي → shape سُلْطَانٌ → سلطنة (السلطان) \rightarrow dim1
dim2

سی بی

الْكَلْمَةُ وَهَا

۲۷۰

shape of S لخواجی از Generalization ای

triangle \approx \sqrt{a} \approx \sqrt{b}

• الک فات دا کلھ ادا ک دایما هست لسايفت عن الک کدة هو

object = object equality 111>g

Note

`child = Parent` "Compilation error"

EX

$$Y = S$$

D W → Error

وہ Error میں سے کسی کو ایسا نہیں کہا جائے کہ وہ ایک ممکنہ نتیجہ ہے۔

Parent is a single Child or Data is

e0 Parent = child ✓

child = Parent x

الآباء

Parent = child

child میں کوئی حاصلات اور Parent اسے حاصل نہ کر سکے اور child 11 is all values اور child میں کوئی حاصلات اور Parent اسے حاصل نہ کر سکے اور child 11 is all values

Parent اسے حاصل نہ کر سکے

`Shape* s;`

لئے address کو دیج
اک child اور اسے shape

+ 0x2000

dim1:10
dim2:20
y=50

`s = & y1;`

`s = & t;`

`s 0x1000`

`y1 0x1000`

dim1:10
dim2:20
Z=100

`s->getArea()`

`s->getArea()`

لہیقہ الہ کنے اور
shape اسے Parent

لہیقہ الہ کنے اور

تیقول اسے ربطیں کہ Function اسے static/Early binding کہا جائے اور Compilation کے وقت میں اسے جسم میں پڑھ لے جائے

call کرنے کا Function

object کو سمجھنے کے لئے pointer کو نوع کے ساتھ میں دیا جائے

`Data type * p;`

اک بیٹا اور علی

نہ کر سکے

Run-time

النقد

`triangle * t;`

`t = &s;`

or `t = new Shape();`

مکر عیف و نیو خواهد
گزیر مفت

invalid

Compilation Error

مینفست اور پرانت سیاور بخواهد Child مینفست اور Parent

Parent ایں کوئاں اور child

Static binding

Parent ایں بخواہ اور child

"late binding" dynamic binding

Function کو کسی دلیل نہیں کہ اس کو کسی دلیل

Parent ایں virtual کو

کوئی پیغام نہیں

is function

run-time کے حسب

بخواہ اور child object کا

virtual بخواہ Parent ایں Function کو

dynamic binding بخواہ ایں اور Pointers کو

child object کا بخواہ ایں ایں Function کو

class shape {

virtual float getArea(); }

};

Shape * s;

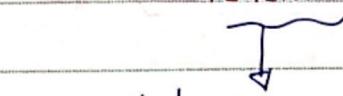
s = new Rect(10, 20);

s->getArea();

کسی الگوریتم کے
rectangle کو

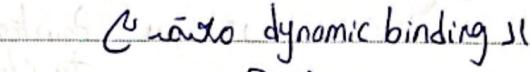
النفی

$\text{Parent}^* \text{s} = \text{new child}()$



into static binding

virtual function



dynamic binding

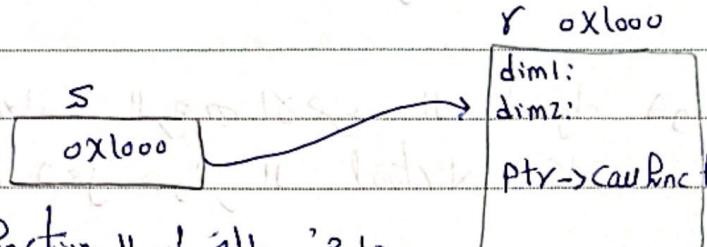
virtual function

virtual function

virtual function

virtual function

virtual function



virtual function

virtual function

virtual function

address

dynamic binding

Parent is virtual function new implement

child is override

object child Parent gives Pointer

child alias s Parent gives reference

parent

override

Parent is override

local function

Goal

الفرقة بير اى (ع) ← Reference from Parent gl Pointer to child
to child

نفس تأثير اى دب اى Pointer to child

لذت ذكر و هست ملحوظ Two alias Name هو Reference
memory له مكان في اى

هذا المكان هو object اى امراء هو reference
من اى حمل حوار او virtual داكل

Concrete class ينبع object فـ اى class اى
له ينبع كامل implementation

لو عابز (ع) Pure virtual Function ←
implementation ينبع من virtual Function

Pure virtual Function
virtual returntype FuncName (...) = 0;

one Pure virtual Function ينبع من class

one has لـ abstract class اى class ينبع اى

reference gl Pointer one has object

Parent * P = new Child();

لودا

Abstract

Parent P; X

ابstract اى

النقط

Pure virtual function \rightarrow class نیوج
 Not overrided \rightarrow implementation \rightarrow کوہاں
 Abstract class نکاح نیوج

Class A {

=

virtual void myFunc() = 0;
 y;

A* a;

a \rightarrow myFunc();

\rightarrow undetermined behaviour

Rabage address \rightarrow L

Class shape {

=

y

Class Rect : public shape {

=

void printArea() {

y =

y;

Shape* S = new Rect();

S->printArea(); "invalid"

ایسے کوہاں shape کی is سے جو دینے والا
dynamic binding کی

o کوہاں کو لئے private member کی سوچ کر کوہاں کو
destructor constructor کی کوہاں کو لئے.

گھٹلی

Parent * arr[3];

element کے 3 element کے array کے
کے child کے Parent کے address پر

Ex shape * t[3];

t[0] = new Rect(5, 6);

t[1] = new Triangle(5, 6);

t[2] = new Circle(10);

float sum = 0;

for (int i=0; i<3; i++)

sum += t[i] -> calcArea();

3

cout << sum;

کے خلیت اور سالنہ (بچے) array

Parent کے 3 child کے constructor کا

composition کے child کے constructor کا

کے destructor کا

child کا

ویسچ رج و هو خالع object

object کے

parent کے

الفہرست

Class A : — B ↗

≡

g

و Private

و Pointer و Private

و صغير من Parent

(نوع الوراثة هنا)

لذلك في الممتاز

أعمى في كل الوراثة

Public

→ Abstract class (C++)

هيئ اسمها باستعمال keyword abstract

و اند Pure virtual function في كل احادي Class

Class Shape ↗
≡
و shape هو pointer

Square و rect في

Class Rect: Public Shape{

≡
Class Square: Public Rect{ ↗
Multi-level inheritance ١٥١

square(): Rect()

≡

Constructor Chaining

و في متعددة

level ↗

Class متعدد ارث ↗ multiple inheritance

نفس الوقت

الآن

shape* s = new square();
 cout << s->getArea();

Function این که کدامیک از
 hierarchy ای overide می‌گویند

shape "virtual" می‌گویند.
 ↓
 rect "override" می‌گویند.
 ↓
 square می‌گویند که این عامل
 "override" می‌گویند

از اینجا virtual ای و کاملاً virtual است.
 اینجا C++ ای و virtuality را بگیرید

Final Keyword in C#

override (یعنی هر دو) (عمل)

by default کهیں virtual function override نیست

virtual کهیں درج نہ کرو

الآن

Day 6

→ static Member

معنی object کو shared یک static member خواهد داشت.
 data segment میں یہ object global ہے۔

Program کا memory میں اسی میں static variable ہے۔

Ex

Class A {

static int c = 0; X

Compilation Error

y;
};

object member ہے۔

المح کرنا

یعنی کل object کا اسی ایسا ایسا ہے۔

بچے۔

Class A {

static int c;

y =

int A::c = 0; ✓

Private Variable

Named class :: static member

ایسا کہ static member ہے۔

Private یک static member کو سب da ہے۔
 also have static function.

کوئی

class A {

 static int c;

 =

 static int getc() {

 return c;

 قرأتكم
 class A {

 ج.

 int A::c = 0;

 سيتم تعيين قيمة

main() {

 A::getc();

}

Non static (أول) و static variable

↳ EX object member (instance Member)

this could be ↳

EX class A {

 int B = 0;

 static int c;

 static int getc() { ↳ Compilation Error
 object member ↳

 return B;

}

;

int A::c = 0;

object Function

↑ will be static member ↓ non static Function ↳

both static و both static ↳

الآن

⇒ Operator overloading

Complex C₁(10,20);

Complex C₂(20,30);

Complex C₃ = C₁ + C₂;

عازم احتمال نفع دارد

ویژه علیغ فعال را

نوع از مقدار ارجاعی ارائه

syntax return type operator (class type - , +)

نحو نسبت

نحو کلاس

نحو این پارامتر

نحو left

\equiv

Symbol

Ex: +, -, *, /, --

this → left

right operand

" "

operator overloading یعنی کجا

؛؛ ؛ ؟ ، * ← is all

، sizeof

new operator یعنی کجا

precedence یعنی کجا
Associativity یعنی کجا

operand یعنی کجا

Ex Complex operator+(Complex c){

Complex res;

res.real = real + c.real;

C₃ = C₁ + C₂.

res.img = img + c.img; C₃ = C₁.operator+(C₂); gl

return res

};

Promotion یعنی کجا

Ex C₃ = C₁ + 4 Constructor یعنی کجا در Complex

error یعنی کجا در این مجموعه داده ای داشتم موجود نباشد Img=0 یعنی کجا

Note

C₃ = 4 + C₁

error

this یعنی کجا در این راستا

گفت

$$C_3 = C_1 + C_2;$$

عملية operator overloading (ج) عاين (ج)

stand alone function

return type operator+(int x, Complex c)

g;

Ex int operator==(Complex r) {

return Real == r.real && Imag == r.Imag;

g

int operator !=(Complex r) {

return !(*this == r);

g

('=') assignment (ج) overloading (ج)

by default ج (ج)

Pointer (ج) shallow copy

هيلجو ميتاورونج نفس اكادميه

Shallow Copy (ج) object (ج) هو هو جو دفع ايد

Void operator=(Complex & r) {

Real = r.Real;

Imag = r.Imag;

g

$$C_3 = C_1 = C_2;$$

فرج ج

Compilation error (ج) void ← void

الآن

١٥) مفهوم المقارنة (عمدات المقارنة) equal operator

Complex operator= (Complex & r) {

real = r.real;

Img = r.Img;

return *this;

object copy
by value

by ref

كى return

copy constructor

ويملاك copy constructor

لوعن two return r

function return مع ارجحه اول

١٦) مفهوم المقارنة لو return by reference

Complex & operator= (Complex & r) {

real = r.real;

مروج ارجاء this, r

Img = r.Img;

local variable

this or return *this;

for Function

بيانو على this

this = temp

بيانو على this

برمجة برمجة Copy Constructor

Complex & operator=

Complex operator++() {

real++;

return *this;

3 void C3 = ++C1

Postfix

Complex operator++(int) {

Complex result = *this;

res++;

return result;

function return by ref

good

Appendix

→ Friend operator overloading

↳ two / Many Parameter

لو عاين

EX

Class Complex {

=

Friend Complex operator+(Complex c₁,); ↗ is
" " ↘ declaration

Complex operator+(Complex c₁, Complex c₂) { definition ↗

Complex t; ↗ is :: operator+ ↗

t.r = c₁.r + c₂.r; ↗ Complex::operator+

t.I = c₁.I + c₂.I;

return t;

};

→ operator overloading insertion

Class Complex {

=

Friend ostream& operator<<(ostream&o, Complex&a);

};

ostream& operator<<(ostream&o, Complex&a) {

o << a.r << " + " << a.i;

return o;

void دا کات

};

Main() {

Complex c(3, 4);

<<) '5' , 200 ملہ

endl گی

int x = 10;

operator (cout, c)

cout << x;
cout << c;

1, 2, 3, 4, 5

Types of Class based on Purpose

Concrete class

→ reusability

Abstract class

Some Concrete Function

" Pure "

interface

All are

Pure Function

① Reusability

② Polymorphism

to Polymorphism

Error

Syntax Error

Compiler

logical Error

result difference

detect by debugging

run-time Error

Mishandle APP

① bad i/p

Errors - UI Programmer

② unavailable Resourc

Run time error → Program crashes, Program stop, abnormaly without completing the execution

Exception → is Nothing but a situation in which we get a run-time error

Exception handle → Giving a programmer message to user informing about the exact problem and also providing him guidance to solve that problem

Goal

try {

تقدر تجيء كثر ①

Catch (...) ;

Catch all تقدر تجيء ②

g =

ترتيب ارجح catch

Catch (...) ;

① child class

y =

② Parent "

لو علست الرئيسي هيل warning

رسالة exception (هنا المخرج)

→ throw to communicate between two function.

throw

- int
- float
- char
- double
- string
- user defined

→ My Class Exception

class myException : public Exception {

char* what();

override

g,

g

→ Function throw Exception سو فونكتشن هيل زوجي

int division (int a, int b) throws (myException) {

if (b == 0) throw myException;

return a/b;

Exception

→ if (b == 0) throw myException;

→ return a/b;

→ if (b == 0) throw myException;

→ return a/b;

→ if (b == 0) throw myException;

→ return a/b;

→ if (b == 0) throw myException;

→ return a/b;

⇒ Inline Functions

Calling by stack frame دوں هستے تکمیل یا

الاستبدال replacement يتم عن الكود .

E-X

Class Test 3

Main() {

Public

Test +

Void Func1(){

```
Cout << "Inline";
```

لوکت هنارکه

inline

void func2() {

لہنہ کو اعلان

t. Punc(s).

+ Funczioni

• 100

سبیٹ

inline سے
calling لے

```
→ void Test::func() { cout << "Non-line"; }
```

inline void cout << "Nonline";

٦

ای دالہ خود داخل ہر class اور function کی مدد سے اس کا استعمال کیا جاتا ہے۔

جس Compiler اور inline کرنے کے لئے ہست و طبعات کرنے کے لئے

-**جیسا کیا** online **کرنا** اتنا تقریب علیحدہ و ہلو نہ ممکن ہے اسکے حنینا۔

① تبنّ ود السرقة لذاتها تقلل منه وقت الاستدعاء المألف Ade

٣) مفيدة للدوار المحرّك والسيارة

مودود الله كانت كيتن حملها `CodeSize` inline مزدوج او

الأقصى

→ structure vs class

class در C++ از struct نیست بلکه struct در C++ از class است.

عادی functions را که در struct داریم در C++ می‌توانیم در class نیز داشت.

functions را که در struct داریم در C می‌توانیم در class نیز داشت.

هر دو همان‌طور است.

→ struct

↳ all things default is public

→ class

↳ all private

→ Friend Function

object که در class داریم فرند function را داریم که در private member را بتوانیم در access کرد. Function در داخل class نیز دارد.

Ex class Test {

private:

int a;

void Func();

protected:

int b;

Test t;

valid ↳ ۳۵ t.a=10; ✓

public:

int c;

function ↳ ۲۷ t.b=20; ✓

Class & friend ↳ t.c=30; ✓

Friend void Func();

g;

g;

گوشی

⇒ Template

→ are used for generic programming
Generalization → write a code work for any data

① Template Function

template <class T>

T Maximum (T x, T y) {
 return x > y ? x : y;

y

template <class T, class R>

void add (T x, R y) {

Cout << x + y;

int sum

~> cout

3

T → int

R → float

add (10, 12.14);

② Template class

template <class T>

class Stack {

private:

T S[10];

int top;

public:

void Push (T x);

T Pub();

y;

template <class T>

void stack<T>::Push(T x)

=

template <class T>

T stack<T>::Pop() {

=

int main () {

stack <int> S;

stack <float> SL;

Good

⇒ Preprocessor

→ instructions to compilers

1 Constant

`#define None var`

Compilation will replace var with None →

Note `#define PI 3.14` warning given compiler is
`#define PI 3 > 3` → PI is being

Ex `#define C cout`

`C << "lo";` Compilation will replace C with cout → `Cout << "lo";`

2 Macro Function

`#define Name(--) ()`

Data type or Parameter will be ↑ ↑ EXPRESSION.

Ex `#define SQR(X) (X*X)`

`int main() {`

`Cout << SQR(5);`
 (5×5) will be replaced by

3 Condition

`#ifdef`

or `#ifndef` value

`#endif`

File guard will be used

`#define name(x) (#x)`

`Cout << name(cahmed);`

will be string Jisne #

string algii #

Ques

→ Namespace

→ are used for removing Naming Conflict

نفس class او Signature ای Function کو یہی نام دے سکتے ہیں۔

Namespace میں 2 فریڈ ایف و ایم (جتنا مددی ایف و ایم)

Ex

```
namespace First { void Func() { cout << "First"; } }
namespace Second { void Func() { cout << "Second"; } }

int main() {
    First::Func();
    Second::Func();
}
```

or

using namespace First;

int main()

↪ Func();

First::Func();

↪ Second::Func();

using namespace ... :: ...

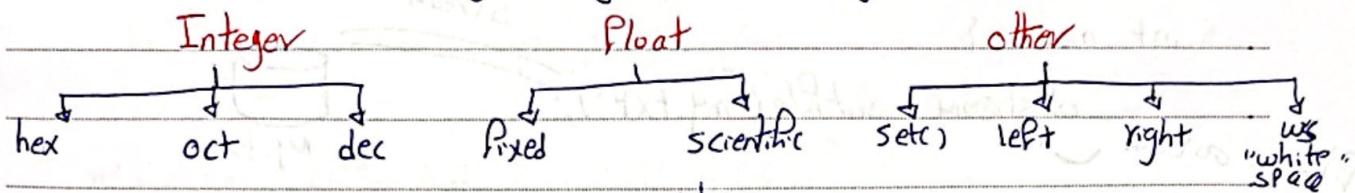
namespace ... ↪

کوئی

النحو

⇒ Manipulator

→ used for enhancing string or formating string



Ex Cout << endl;

Cout << "n";

Cout << hex << 163;

Cout << Fixed << 125.73;

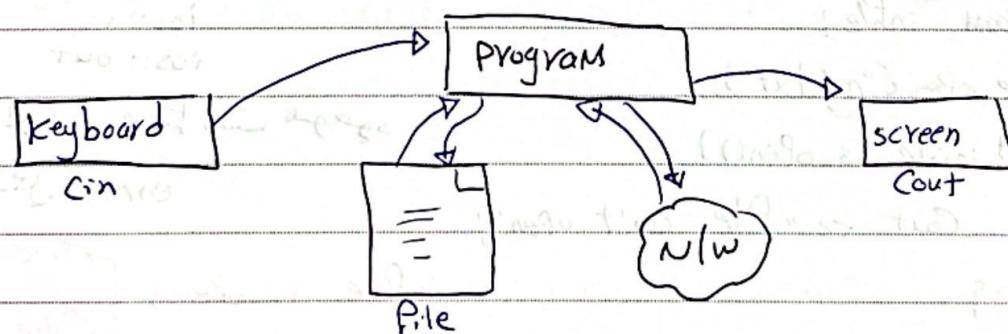
Cout << setc(b) << "Hello";

Cout << lo << ws << ro;

⇒ Streams

→ Flow of data

→ streams are used accessing the data from outside the program that is for external sources of destination.



ex

io stream

input file stream

"ifstream"

ostream

output file stream

"ofstream"

Goal

⇒ File Handling

```
#include <fstream>  
int main() {
```

File is initialised → ofstream coutfile("my.txt"); my.txt

`ofstream outfile (" ",)`
Path ↑ →ios::app "append"
→ios::trunc "truncate" default

فہرست کھوئی اور back up file write

لوار Pile هنگر و ادیم هست موجود در ofstream هست

```
outfile << "Hello" << endl;  
" " << zs << " ";  
outfile.close();
```

my.txt
Hello
zs

```
#include <iostream> // 引入 iostream 头文件  
ifstream infile(" ", );
```

int main() {

```
ifstream inFile;
```

if(!infile.is_open())

```
cout << "File can't open";
```

else if

string str,

int x;

→ infile <>> sty;

$$11 \gg x,$$

" - Close "

سیدی ۶

ଶ୍ରୀମତୀ

```
inFile.open(" ", )
```

Path

`ios::in`
`ios:: out`

لوار ۲ File هست مجموعه

error yes

if file is open {
 // do something
}

الملف المفتوح Open file

الفقرة 8 (فتح File) ۱

→ true

inFile.eof() {

Org file 11-2016 Jpg

⇒ Serialization

→ store to File and again retrieve from a File

وسيتم تخزينه في ملف وستعاد لاسترجاعه كـ object

deserialization ↗

تحويل البيانات داخل object المخزنة كـ

Program ↗

two type of files

Text ↗

"Human readable"

Ascii برمجيات

Binary File ↗

Machine readable

Mode ios::binary

func read(), write()

Ex 23 text

ASCII 50 51 ASCII ↗

0110010 0110011 binary ↗

→ More space

binary ↗ Ascii ↗

→ Paste ↗

less Space ↗

Ascii برمجيات

binary 00000000 00000000 00000000 10111111

أو حاول لبيانه كـ

sunk character ↗

أو فحص ملائمة بـ

Ex Boost.Serialization

Cereal

Protobuf

Goofy ↗

\Rightarrow auto keyword

متاتا (انواع دیکست نوع ار Data type) موده
تیکسیت مها لو معرفت نوع ار " " ایکی هنگام

ex auto x = 2 * 5 - 1 + '0';
double and i, j

\Rightarrow dec type (variable)

او هر دو صنیع رنگ دار Data type یا جنس نام

`decl type(var) name = value;`

Ex $\int x \, dx = \frac{1}{2}x^2 + C$

`datatype(X) Y = Z0;`

int \leftarrow X \wedge \neg $\exists x \forall y \neg$ $\exists z$

⇒ Final keyword

• لوحت معاد class يعَف هفترست او، ت هنَّه
overide هفتر اعلان Function داخل او class هفت هفت

virtual function

virtual return type Name (--) Final {

3

class A final {

一

5

Class B: Public API error

32

الأخضر

⇒ lambda EXPRESSION

→ unnamed Function

[Capture-list] (Parameter-list) → return type { body } ;

2

لهمه مفروض حاصل

replace parameters

value (نها. كما في)

lambda ()

by reference (عن طريق ارجاع) ←

مع الأرجاع

⇒ Calling "Parameter"

[] () → int f -- y ();

Parameter (أرجاع)

Ex int main()

[] () { cout << "Hello"; y(); }

[] (int x, int y) { cout << x + y; y(10, 5); }

Function و TS passing args to lambda !!

Ex auto P = [] () { cout << "Hello"; y(); }

f();

Ex int a = 10, b = 5;

[a, b] () { cout << a << " " << b; y(); }

using args, lambda !!

لهمه مفروض (عن طريق ارجاع) !!

by reference args lambda

Ex template <typename T>

void Fun(T p) { }

P();

Parameter S lambda

y

Good

⇒ Ellipses

→ used for taking variable number of arguments
in Function

Parameter list is used in Function like this

Ex int sum(int n, ...)

va-list list;

va-start(list, n);

int s = 0;

for (int i=0; i < n; i++)

s += va-arg(list, int);

va-end(list);

return s;

sum(10, 20, 30); $\rightarrow n = 3$

sum(5, 9, 4, 23, 7); $\rightarrow n = 6$

va-list list;

va-start(list, n) \rightarrow jdk1, jdk2

va-arg(list, type) \rightarrow jdk1, jdk2

↳ return element one by one

va-end(list)

Goal

⇒ Smart Pointer

→ automatically manage memory and they will locate the object, when not used, when the pointer is going out of scope automatically it will deallocate the memory.

Ex

func()

Rectangle* P = new Rectangle();

3

P

R

main {

while(1) {

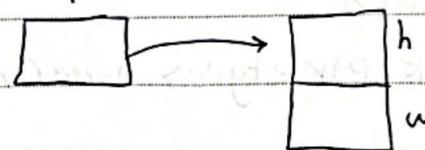
func();

مثلاًSmart Pointer يُخْبِرَ بِمَنْ يُحْتَفِظُ بِهِ

لذلك إذا تم تخلص من الـ pointer فلن يُعْلَمُ بِهِ ذَي الْ اِنْتِهَا

Memory leak يُعْلَمُ بِهِ الذُّي لَمْ يُعْلَمُ بِهِ ذَي الْ اِنْتِهَا

وهو يعني أن الـ pointer لم يتم إصداره



garbage collector يُعْلَمُ بِهِ الذُّي لَمْ يُعْلَمُ بِهِ ذَي الْ اِنْتِهَا

Smart Pointer يُعْلَمُ بِهِ الذُّي لَمْ يُعْلَمُ بِهِ ذَي الْ اِنْتِهَا

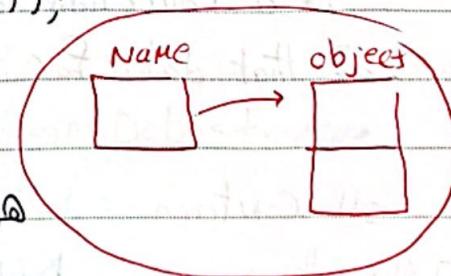
II unique_ptr

unique_ptr<type> name(new Class());

مثلاًSmart Pointer يُعْلَمُ بِهِ الذُّي لَمْ يُعْلَمُ بِهِ ذَي الْ اِنْتِهَا

Pointer يُعْلَمُ بِهِ الذُّي لَمْ يُعْلَمُ بِهِ ذَي الْ اِنْتِهَا

هو يعني فهم كل ملكية الملفات

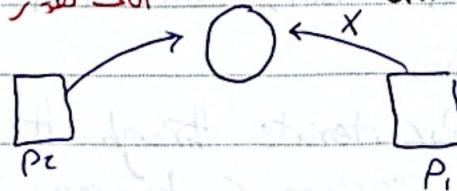


unique_ptr يُعْلَمُ بِهِ الذُّي لَمْ يُعْلَمُ بِهِ ذَي الْ اِنْتِهَا

object يُعْلَمُ بِهِ الذُّي لَمْ يُعْلَمُ بِهِ ذَي الْ اِنْتِهَا

P1 → S0, P2 → P1

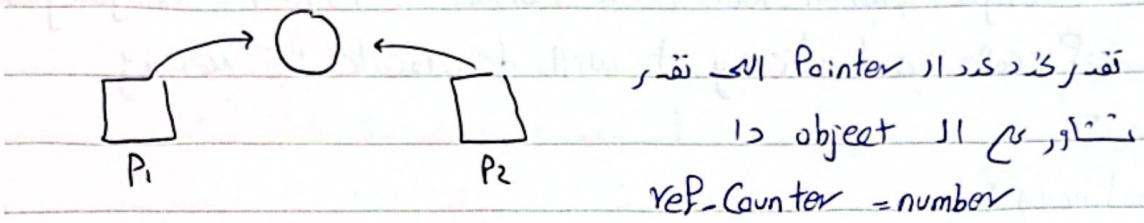
P2 = move(P1) → P1 → null



good

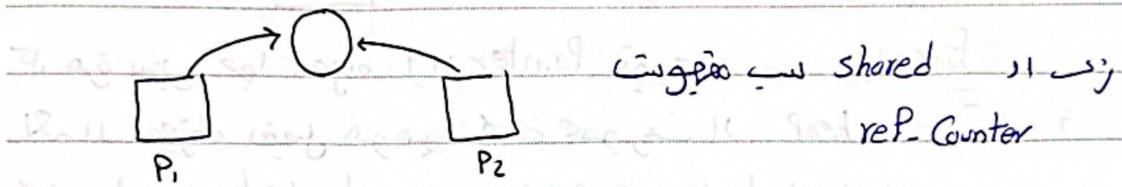
2) Shared_Ptr

`shared_ptr<type> Name(new class());`



3) weak_ptr

`weak_ptr<type> name(new class());`



⇒ STL

→ standard template library

→ Built-in Data Structure

is a Collection of data and the arrangement
of that data for its efficient utilization

Space Time

STL contain

⇒ Algorithms built-in algorithms or functions

⇒ Containers

→ Contains collection of data structures

⇒ Iterators

→ For iterate through the collection of values

Goal — "Access Containers."

Ex Algorithms

Search - sort - binary search - reverse - Concat - Copy
 union - intersection - Merge - heap - etc.

Ex Containers

vector - list - Forward list - deque - priority queue - stack
 Set - Multiset - Map - etc.

1) vector

→ self Managed Array

→ dynamically manage the size of array

→ Push_back()

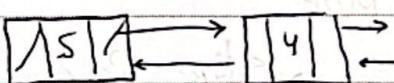
→ Pop_back()

→ insert()

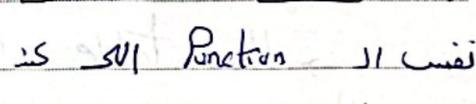
→ remove()

→ size()

→ empty()

2) list

→ doubly linked list



is STL Function

is also vector

→ Push_front()

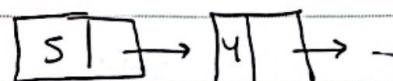
→ Pop_front()

→ front()

→ back()

3) Forward_list

→ Single linked list



list is Function

4) Priority Queue

→ heap Data Structure

→ Push()

→ Pop()

→ empty()

→ size()

graph

5) deque

→ same as vector

الفرفت داير (وذر اهسف و ادف معه اي اجياء)

6) stack "LIFO" last in First out

7) set

→ unique element

→ No duplication

8) multiset

→ same set but have duplicate

9) map

→ store pair <key: value>

→ Hash table & "unique key"

10) multimap

→ same as map

يسمح بـ values متعددة لـ key او key يربط بـ متعدد values

→ iterator

Container type <Datatype> :: iterator itr;

like a pointer.

→ Functions

begin()

end()

reverse

rbegin()

rend()

address تبرمج collection

False, true, gl

graph

Ex

```
#include <vector>
int main() {
    vector<int> v = {10, 20, 30, 40};
    v.push_back(25);
    v.push_back(40);
    v.pop_back();
    for (int x : v)
        cout << x;
```

```
vector<int>::iterator itr;
for (itr = v.begin(), itr != v.end(); itr++)
    cout << *itr;
```

#include <Map>

```
Map<key, value> name;
```

$\text{itr} = \text{name}.find(\text{key}) \rightarrow \text{return pair}$ iterator \Rightarrow $\text{pair} \leftarrow \text{value}$

$\text{name}.insert(\text{pair} \leftarrow (\text{key}, \text{value}))$

$\text{type key} \uparrow \quad \uparrow \text{type value}$

```
Map< , >::iterator itr;
```

$\text{itr} \rightarrow \text{first} \rightarrow \text{return key}$

$\text{itr} \rightarrow \text{second} \rightarrow \text{return value}$

Ques