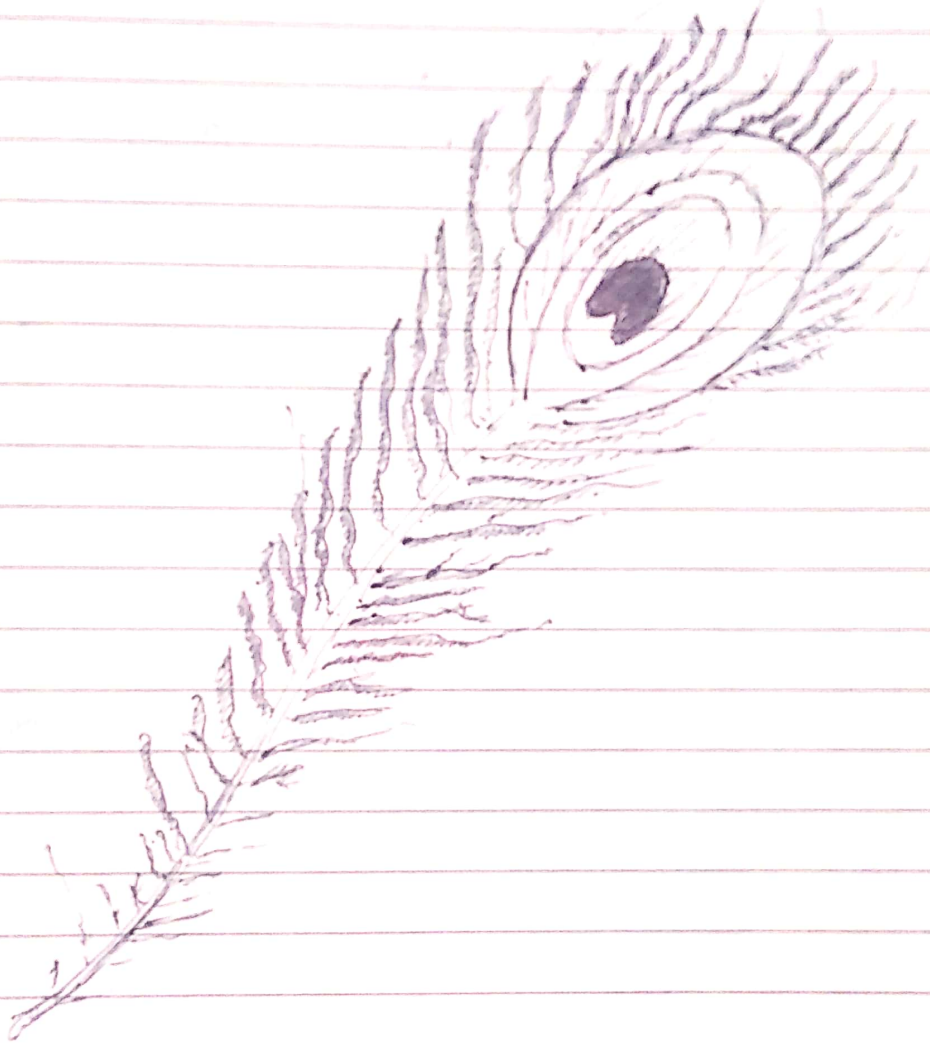


Pointers

Q. Why do we need pointer?

it Stores address in Hexadecimal number
0x7ffe59f3feb.

Variable stores address of other variables.
P = &i

int *p = &i; → int type ka address

P is a pointer of an integer → which stores address of i;

Cout << p << end → address
Cout << *p << end → value

Node
64 bit
int → 4by

int *q = p;

it means *q *p i
same will print same

Concept

→ When address is provided to pointer then it points garbage address and it also Contain garbage value as well as

→ If you not initialising value to pointer then align it null value.

Pointer Arithematic

Size of all pointer is same (8 byte, 4 byte)

Operators on pointer

1.
```
int i = 10
int *P = &i;
Cout << P << end;
  P = P + 1;
Cout << P << end;
```

→ +4 byte

P+1 → next integer → Point to next add

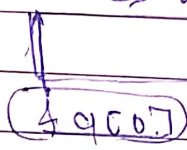
P-1 → previous integer

↳ -4 byte

* Array & Pointer

```
int q[10];
```

↳ name — for — Three ↓ 40 bytes




```
int main ()
  int q[10];
  cout << q << end;
  cout << &q[0] << end;
    q[0] = 5;
    q[1] = 10;

  Cout << *q << end;
  Cout << *(q+1) << end;
```

→ 1st element
* a address

→ 1st element

→ next element

i[a] → *(i+a)
a[i] → *(a+i) > same

Camlin Page
Date / /

* ① size of
② 4 op.

Concept

In case of array → a[10]
cout<<a
↳ address of 1st elem

size of (a)
↳ whole size → 40 byte

int *p = &a[0];
↳ add 700

p → add at 890

now can change

*p = &a[3]
↳ +8 byte (708 byte)

a++ not allowed

Array
(a = a+3) → a is correspond to other memory
↳ can't change

* array can't be re assigned.

Date / /

Character & pointers

* it behaves diff in character

```
int main()
{
    int a[] = {1, 2, 3};
    Char b[] = {"a b c"};
    cout << a << endl;
    cout << b << endl;
```

↳ a b c print karega null character dekh k ruk jayega

```
Char * c = &b[0]
cout << c;
```

↳ phir se abc print karega jab tak null char na miley

```
Char c1 = 'a';
Char *pc = &c1;
cout << c1 << endl;
cout << pc << endl;
```

* array jab initialize karte hai Tab. ek temporary memory create hoge and usma wo temporary memory ki copy bane.gi

* pointer mai aisa kuch nahi hota vo temp memory ko point karte hai

Pointer & functions

Print f^n

```
void print (int * p)
    {
        cout << *p << endl;
    }
int main()
    {
        int i = 10;
        int *p = &i;
        print (p);
    }
```

Address source

```
void increment (int *p)
{
    (*p)++;
}
```

↑ value change

main

p → [700]

increment Po...

p → [704]

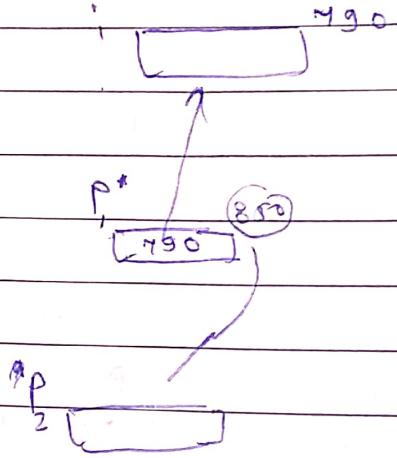
```
    cout << p << end,
    increment Pointer (p);
    cout << p << endl;
}
```

Array passed as pointer

```
int sum (int *a, int size)
    {
        int ans = 0;
        for (int i = 0; i < size; i++)
            {
                ans += a[i];
            }
        return ans;
    }
int main()
{ int a[10];
  cout << sizeof(a); cout << sum(a+3, 7) << endl;
```

Double Pointer

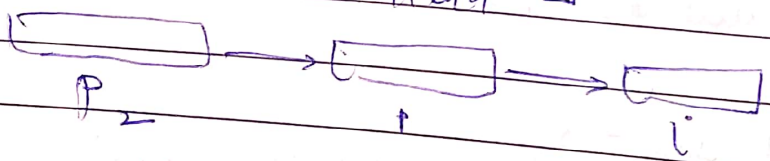
```
int *p = &i
```




```
int **p2 = &p
```

```
int main()
{
    int i = 10;
    int *p = &i;
    int **p2 = &p;
    cout<< p2 <<endl;
    cout<< *p2 <<endl;
}
```

Same



Dynamic Allocation

Type casting

Why we write pointer with *?
like why we have to initilize datatype

like - why int * p;
why not pointer * p;

* because we have to read byte and how to interprate

```
int i = 65;
char c = i;
cout << c << endl;
```

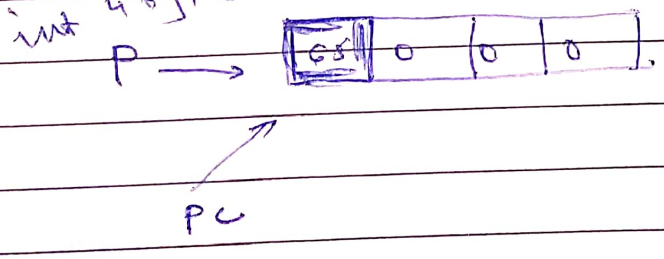
implicit

```
int * p = 4i;
char * pc = p;
```

explicit

```
int * p = 4i;
char * pc = (char *) p;
cout << *p << endl;
cout << *(pc+1) << endl;
cout << *(pc+2) << endl;
cout << *(pc+3) << endl;
```

→ null



Reference & Pass by reference

Reference Variable

```
int main()
{
    int i=10,
    int &j=i;   ⟶ j mai i ka address
    i++;          store so ju change
    cout<<i<<endl;   I mai vo i mai vsq
    j++;             versa
    cout<<i<<endl;
    int k=j;
    cout<<k;

    int &j;
    j=i      ✗  wrong
```

∴ when declared you have to also initilized

Pass by reference in fn.

```
Void increment(int &n)   (2) shared same memory
{
    n++;
}

main()
{
    increment(i)
    cout<<i<<endl;
}
```

* benifit → . value replace kar sakte hai
→ same memory use kar sakte hai.

What if? (Return by reference)

```
int & f(int n)
{
    int a = n;
    return a;
}
```

Created local var.

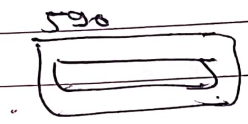
→ fn created memory for some time

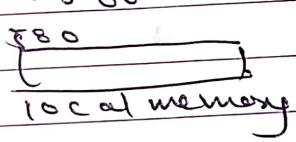
Similarly fn se int pointer.

```
int * f2() {
    int i = 10;
    return &i;
}
```

Created local var.

i → 580






```
int main()
{
    int &k = f(i);

    int *p = f2();
}
```

→ it may clear memory after fn call.

Basically,

Ek fn ok jise bhi local variable ka reference ya pointer return karwayegey to problem hoge because it will vanished.

Dynamic Memory Allocation

bad Things

```
int a[20];
int n;
cin>>n;
int a[n];
```

→ fixed size array
→ decided on runtime

What DMA do → drawback door

* Whenever we allocate memory

Stack → when program starts.

Heap → huge memory

When you write

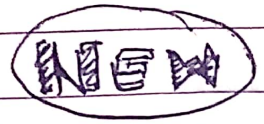
```
int i =10;
int a[20]
int a[2000];
```

→ stored in stack;

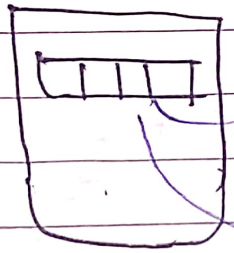
In Heap → we can allocate memory in runtime.
→ this dynamic allocation

How D M A works

HA! HA! HA!



new int



Heap

→ it will creat 4 byte in Heap

→ we donot give name to new.

↓

Ye memory ka address lauk dega.

then hum use? pointer mai save kar lege

like int *p = new int;

*p = 10

→ if you wont to change

step

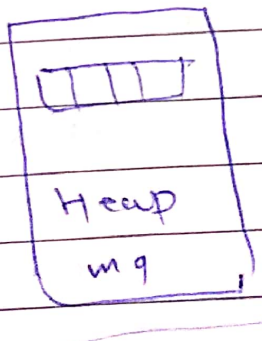
store karleyg

Address aayg

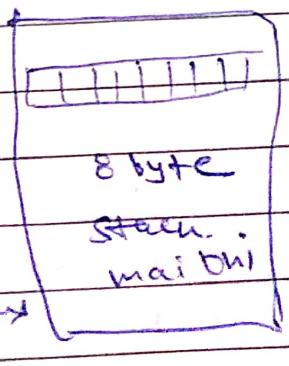
int *p = new int;

*p = 10;

cout<<*p<<endl;



Heap mg

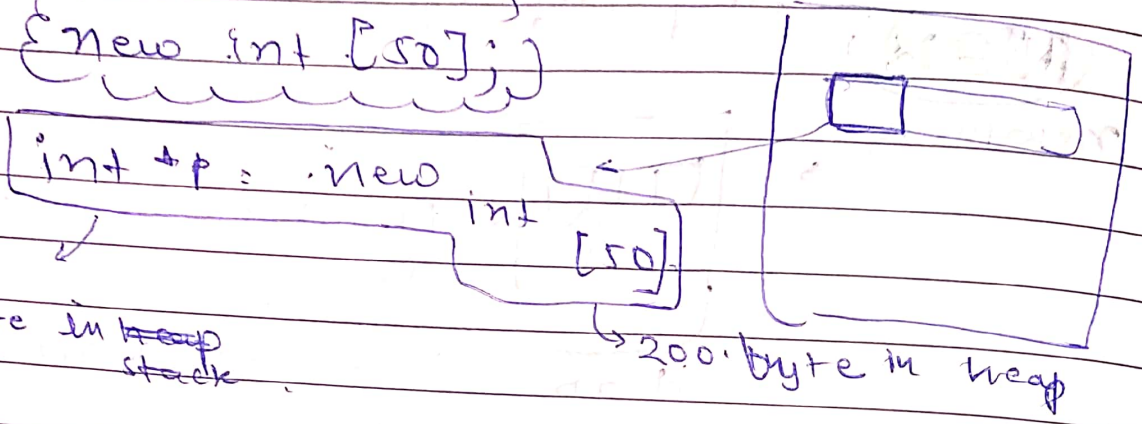


8 byte stack mai bhi

How to create Array dynamically

new int [50];

int *p = new int [50];



→

```
int n;
cout << " ____ . "
cin >> n;

int * p2 = new int [n];
for (int i=0; i<n ; i++)
    {
        cin >> p2[i];
    }
int max = -1;
for (int i=0; i<n ; i++)
        if (max < a[i])
            {
                max = a[i];
            }
        }
    cout << max << "\n";
```

Camlin Page
Date / /

* Major diff. b/w static & dynamic memory

ex → int main() {

```
while (true)
{
    int *p = new int;
}
```

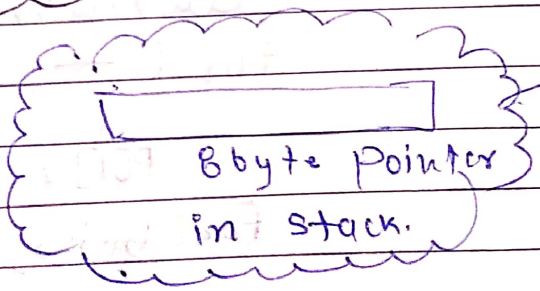
```
while (true)
{
    int i = 10;
}
```

↳ during this iteration the memory will creat in stack and automatic delete

→ But when this called the memory is not deleted you have to delete the memory by

```
delete p;
```

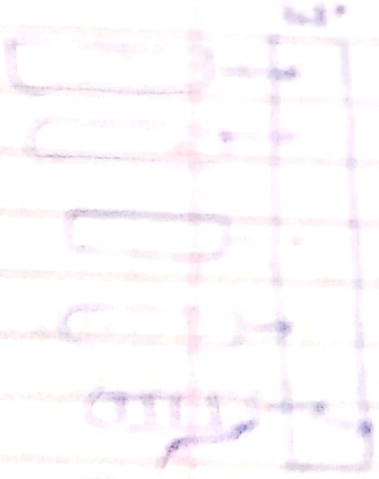
We do not deleted p. We deleted the element which p pointing



*this automatic deleted

by delete p. it will delete the meory of in DH

What About 2-D Array ?



```
int main()
{
int** p = int new * int[n];
int m,n;
cin>>m>>n;
for ( — i<n ; i++)
p[i] = new int[n];
for( int i=0; i<n; i++)
{
cin>> p[i][j];
```

How much memory allocate in (I, 10)

Q. How to deallocate the memory ?

```
for (int i=0; i<m; i++)
    {
        delete [] P[i];
    }
delete [] P;
}
```

↑ array <- use and ader array ki memory delete

Why dow't we create like int[m][4] in pointer;

Macros & Global Variables

1. #define

ex:- #define PI 3.14

↳ closent take storage memory

↳ Compile Hone se pehle he ye (PI) ko find kar k 3.14 Kar dega.

no extra memory.

2. Global Variables.

int a; → globally defined

if value in one fn change then the value changed in all fn.

(X) load to use

```
void g() {
    a++;
    cout<< a <<end;
}
void f() {
    cout<< a <<end;
    a++;
    g();
}
int main()
{
    a=10;
    f();
    f();
    cout<< a <<end;
```

Inline fn;

(1) Tiritary operator

int c = (a>b) ? a : b.

agar True to c mai a
agar false to c mai b.

```
inline int max (int a, int b)
{
    return (a>b) ? a : b;
}
int main()
{
    int a, b;
    cin >> a >> b;
    int c = max(a,b);
    int x, y;
    x = 12;
    y = 34;
    int z = max(x,y);
```

Copy Copy.

* means hoye fn nahi hai pura code yahi fn mein mai aayega;

* How to make four "{" from single {4

→ make it default argument

Note :- You can only start making arguments default from right side

```
int sum2(int a, int b=0, int c=0, int d=0)
{
    return a+b+c+d
}
```

What it mean?

```
int main()
{
    int a[20]
    cout << sum(a, 20) << endl
```

(1)

```
sum(int a)
{
    return a+0+0+0;
}
sum2(int a, int b)
{
    return a+b;
}
sum3(int a, int b, int c)
{
    return a+b+c;
}
```

Making Arguments Optional

↳

Constant Variables

```
int main()
{
    const int i = 10;
    int j = 12;
    const int &k = j;
    k++;
    j++;
}
```

we can also write int const

Wahi hi wahi initilize karo

Const lagga deya matlab only read kar sakte bhiya!

Storage constant nai rahi only Path constant.

Pointers

```
int const i = 10;
int cons *p = 4i;
int j = 12;
int const *p2 = 4j;
const *p2 <<endl;
j++
cout << *p2 <<endl;
```

Recursion

fⁿ calling itself

let,

$n! = n * n-1 * n-2 \cdots$

$n! = n * (n-1)!$

fact(n) = n * fact(n-1)

Program for factorial

```
int factorial (int n)
  {
    cout<< n <<endl;
    if(n==0)
      {
        return 1;
      }
    int smallOutput = factorial (n-1);
    return n * smallOutput;
  }

int main ()
  {
    int n;
    cin>>n;
    int output = fac(n);
    cout<<output<<endl
```

Camlin | Page
Date / /

Principle of Mathamatic Induction (PMI)

To prove some facts.

Let,

some fact of $f(n)$ is true $\forall n$.

PMI says,

① Prove $f(0)$ or $f(1)$ is true (Base case)

~~②~~ Assume Induction hypothesis (Assume) $f(k)$ is true.

③ Induction step (using ② prove $f(k+1)$ is true)

Prove sum of n natural numbers $\frac{n(n+1)}{2}$

① Base case $f(0)$ i.e $\Sigma 0 = 0$ (LHS)

RHS $\frac{n(n+1)}{2} = 0$

② $\Sigma 1 = 1$ LHS

$\frac{1(1+1)}{2} = 1$ RHS

③ Induction hypothesis $\Sigma k = \frac{k(k+1)}{2}$

$\Sigma k+1 = \frac{(k+1)(k+2)}{2}$

$k+1 + \Sigma k = \frac{(k+1)}{2} + \frac{(k+1)(k)}{2}$

$\left(\frac{k+1}{2}\right)(k+2)$

Fibbonacci Number

0 1 1 2 3 5 8 13 21 34

fib(n) = fib(n-1) + fib(n-2)

Base Case : Prove f(0) on F(1) is true

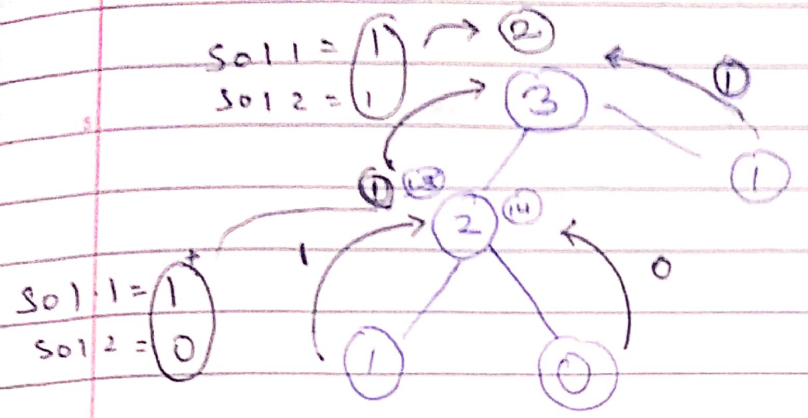
IH : Assume f(i) is true ∀ i ≤ k

Ic : Use (2) to prove f(k+1) is true

```
int fib(int n)
{
    if (n==0)
    {
        return 0;
    }
    if (n==1)
    {
        return 1;
    }
    int smallOutput1 = fib(n-1);
    int smallOutput2 = fib(n-2);
    Return smallOutput1 + small2;
}

int main()
{
    cout << fib(3) << endl;
}
```

Camlin Page
Date / /

dry run



Q Check if array is sorted or not use Recursion?

```
bool is_sorted (int a[], int size)
        if (size == 0 || size == 1)
            {
                Return true;
            }
    if (a[0] > a[1])
            {
                Return false;
            }
    bool isSmallestSorted = is_sorted (a+1, size-1)
            {
                Return isSmallestSorted;
```

dry run

true ← 2 | 3 4 5 6 (5)

true ← 3 | 4 5 6 (4)

true ← | 4 5 6 (3)

true ← 5 | 6 (2)

true ← | 6 (0)

Camlin Page
Date / /

Same problem from 9+1

```
bool isSmaller (9+1, size-1)
{
    if (! is smaller out)
    {
        return false;
    }
if (9[0] > 9[1])
    {
        Return false;
    }
else
    {return true;
    }
```

Q1 Problem ① (1st index) by Recc.

```
int index ( a*, n, x)
    {
        if (n==0)
          {
            return -1;
          }                                  > base case
        if ( a* == x)
          {
            return 0;
          }

    int ind = index ( a+1, n-1, x);

        if ( ind == -1)
            {
              return -1
            }
        else
            { return ind+1;
            }

int main()
  { int i, n, x, k;
    cin >> n;
    for (i=0; i<n; i++)
      {
        cin >> a[i];
      }
    cin >> x;
    cout << index(a,n k);
```

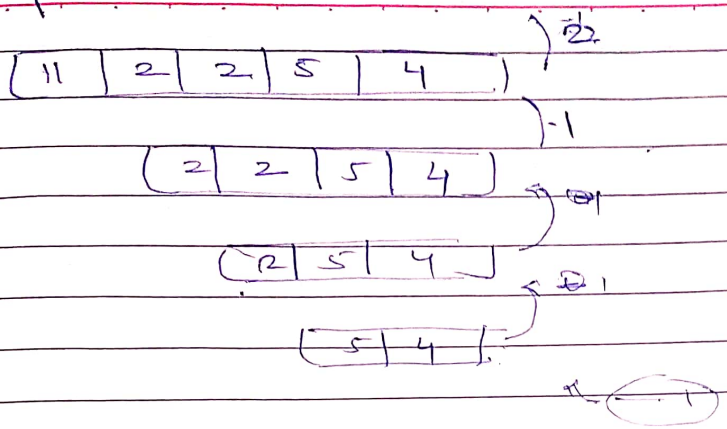
Problem ② Last index.

Same process followed but Recc is in second step then small cac per.

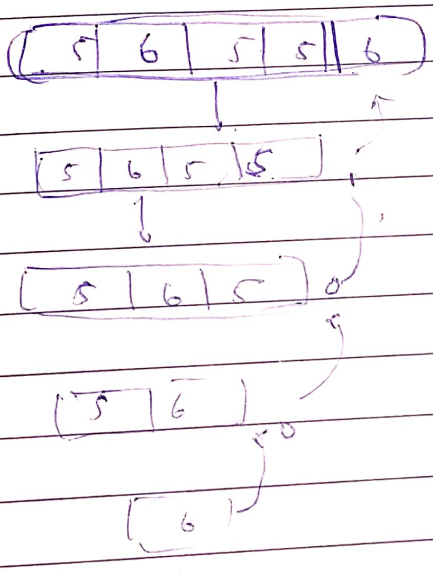
```
int index(int *a, int n, x)
    {
        base case
        if( n == 0)
            {
                return -1
            }

        int ind = index(a+1, n-1, x)
            {       return ind+1;
        if(*a == x)
            {
                return 0
            }
        else
            { return -1
            }
        if (ind == -1)
            {
                if(a* == x)
                    {
                        return 0;
                    }
                else
                    { return -1;
                    }
            }
        Return ind+1;
```

Camlin Page
Date / /

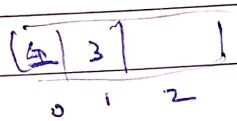
by sum

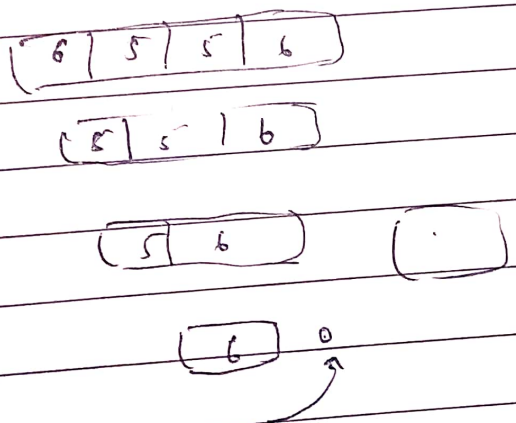


Q # Return all index of given num in array. also in sorted form



size	x	output
5	5	0 2 3 (size 3)
	6	1 4 (size 2)

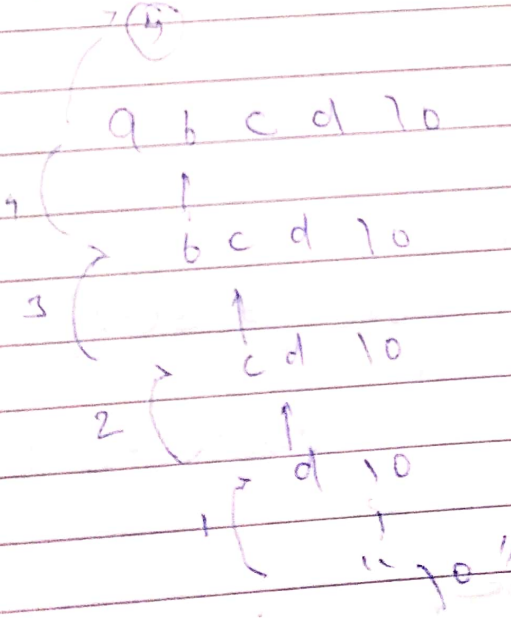




Q. finding length of string using reccursion

```
#include <---->

int length (char s[])
    {
        if (s[0] == '\0')
        return 0;
    }
    int small string length = length(s+1);
    return 1+small stringlength;

int main ()
    {
        char str[100];
        cin >> str;
        int l= length(str)
        cout << l << endl;
    }
```

dry run



Remowe Duplicates from String Using Recc.

```
#include
using —

int length (char s[])
{
    if (s[0] == '\0')
        {
            return 0;
        }
    int small string length = length (s+1);
    return small string;
  else{
  }
Void duplicate (char s[])
    {
      l = length (s[])
      if (l < 1)
        {
            return 0;
        }
      else if (s[0] != s[1])
            {
                duplicate (s+1);
            }
      else { int i = 1;
             for (i; s[i] != '\0'; i++)
                { s[i-1] = s[i];
```

```
    s[i-1] = s[i];
    duplicate(s);
}

int main()
{
    int s[100];
    cin>>s;
    duplicate(s);
    cout<<s;
}
```

Dry run

k	a	a	a	b	c	c	d	d	s	\0

l = 10

a	a	a	b	c	c	d	d	s	\0

a	a	b	c	c	d	d	s	\0

a	b	c	c	d	d	s	\0

b	c	c	d	d	s	\0

c	c	d	d	s	\0

c	d	d	s	\0

d	d	s	\0

d	s	\0

s	\0

k	a	a	a	b	c	c	d	d	s	\0

k (a a a b c c d d s \0

k (a a b c c d d s s \0

↳ a a b c c d d s \0

k (a b c c d d s \0)

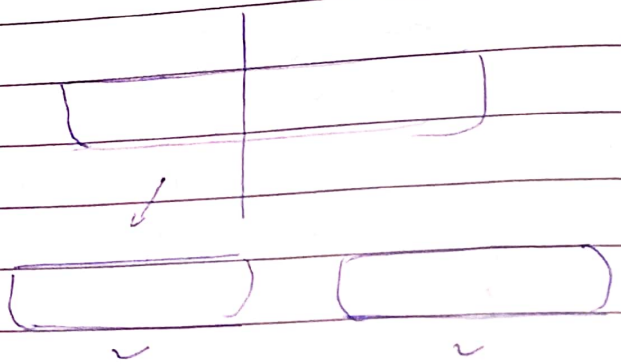
k a b (c c d d s \0)

k a b c d s \0

k a b c d s \0

Merge Sort And Recursion

base case → 1 → 0 → already sorted return
1



Or

(aadhe aadhe pai recursive)

How it works?

```
Void merge sort (int a[], int si, int ei)
    if ( si >= ei)
    {
        return;
    }
    int mid = si + ei
              2
    for ( i = si; i < mid; i++)
    {
        if ( a[i] > a[i+1])
        Swap
```

```
void smaller (int a[], int si, int mid)
{
    for(int i=si; si <= mid; si++)
    {
        if (a[si] < a[si+1])
            swap (a[si], a[si+1])
    }
}

void mergesort (int low, int high)
{
    if (low < high)
    {
        int middle = (high + low)/2;
        mergesort (low, middle);
        mergesort (middle+1, high);
        merge (low, middle, high);
    }
}

Void merge ( int low, int middle, int high)
{
    for(int i = low; i <= high; i++)
    {
        helper[i] = number[i];
    }

int i = low;
int j = middle + 1;
int k = low;
```

```
while (i <= middle && j <= high)
    if (helper[i] <= helper[j])
        {
            number[k] = helper[i];
            i++;
        }
    } else {
            number[k] = helper[j];
            j++;
        }
    k++;
}

while (i <= middle)
    {
        numbers[k] = helper[i];
        k++;
        i++;
    }
```

Quick Sort

| 5 | 9 | 8 | 3 | 2 | 7 | 1 |

| 3 | 9 | 8 | 5 | 2 | 7 | 1 |

| 3 | 1 | 2 | 5 | 8 | 7 | 9 |

Sort Sort.

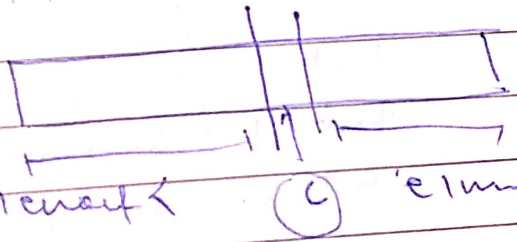
1st Void qs(int arr[], int si, int ei)

{

base case (si, ei)

Return

c = partition(a, si, ei)

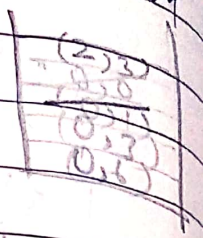


qs(a, si, c-1)

qs(a, c+1, ei);

}

Q Understanding Mergesort using recursion



Code

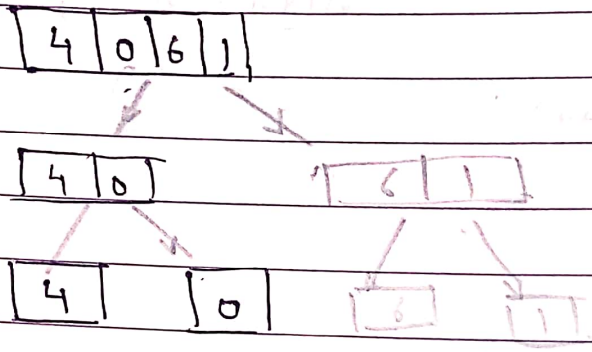
```
#include <bits/stdc++.h>
using ——.

void mergesort ( int q[], int low, int middle, int high)
{
    int helper[n];

    for ( i = low ; i <= high ; i++)
    {
        helper[i] = q[i];
    }
    int i = low;
    int j = middle+1
    int k = low;
while( i<=middle && j <= high)
{
    if ( helper[i] <= helper[j])
    {
        q[k] = helper[i];
        i++;
    }
else
    { q[k] = helper[j];
      j++;
    }
    k++;
}
```

```
int main()
{
    int n;
    cin>>n;
    int a[];
    for(int i=0; i<n; i++)
    {
        cin>>a[i];
    }

    mergesort(a[], low, high);
    for(i=0; i<n; i++)
    {
        cout<<a[i];
    }
}

low = 0;
high = n-1;

while(i<=middle)
    {
        a[k] = helper[i];
        k++;
        i++;
    }
}

void merge sort(int a[], int low, int high)
    {
        if(low<high)
            {
                int middle = low + (hig-low)/2
                mergesort(a, low, middle);
                mergesort(a, middle+1, high);
                merge(a, low, middle, high);
            }
    }
```

Camlin Page
Date / /

Dry run

4	0	6	1	5	2	3

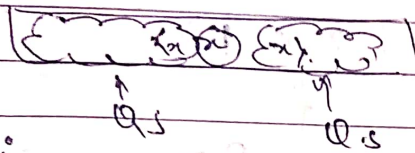


(0,0)
(0,1)
(0,2)
m.s (0,6)

2, 2, 3
(3, 2)
2, 3
(0, 7)
(0, 6)

Algorithm for Q. sort

is k around partition



take si, ei

① void qs (int a[], int si, int ei)

base case.

0 or 1 → sorted.

partition (a, si, ei) → x ko c pai

→ count of smaller elements =
si + count_smaller_element.

```
void qs( q[], si, ei)
{   int n = sizeof(q)/sizeof(int)
    if (n < p)
    {
        Return 0;
    }
}
int c = partition (q, si, ei)
    qs(q, si, c-1);
    qs(q, c+1, ei);
}

void Partition(int q[], int si, int ei)
{
    for (i = 0; i <= ei; i++)
    {
        int k = 0; int m;
        for( q[i] < q[k])
        {
            m = count++;
        }
    }
    swap(q[si], q[si+m])

i = si;
v = ei.
```

```
If (a[i] < x)
    {
      i++;
    }
elseif (a[j] > x)
      {
        j--;
      }
else  {
        swap(a[i], a[j])
```

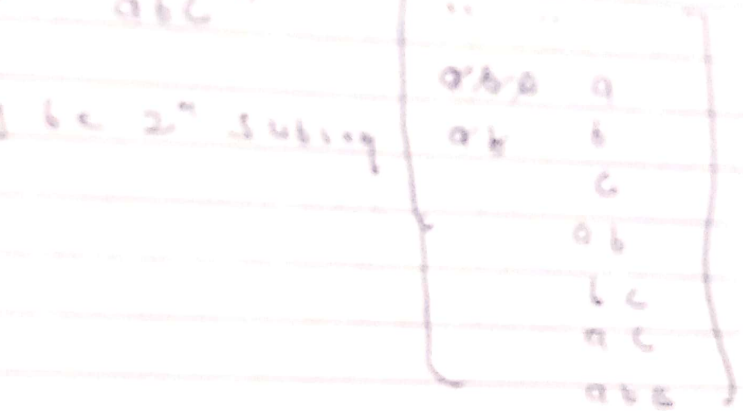
Strings In Recc

- Substring()
  S.substring(2,3) <<
- S.find()
  S.find("cle")

V.2 imp

Q. How to find all subsequence of string

ex → "abc"



may be 2^n subsq

```
void Subs(String input, String output[])
{
    if(input.equals(""))
    {
        output[0] = "";
        return 1;
    }
    String smallOutput = Subs(input.substring(1), output);
    String smallString = input.substring(0,1);
    for(int i=0, i<smallOutput; i++)
    {
        output[i+smallOutput] = smallString + output[i];
    }
}
```

```
int main()
    {
        String input;
        cin>>input;
        String * output = new String[1000];
        int count = Sub(input, output);

        for(int i =0; i<count ; i++)
            {
                cout<<output[i]<<endl;
            }
```

Q. Keypad Problem

2 → "abc"
3 → "def"
4 → "ghi"
.
.
.
9 → "wxyz"

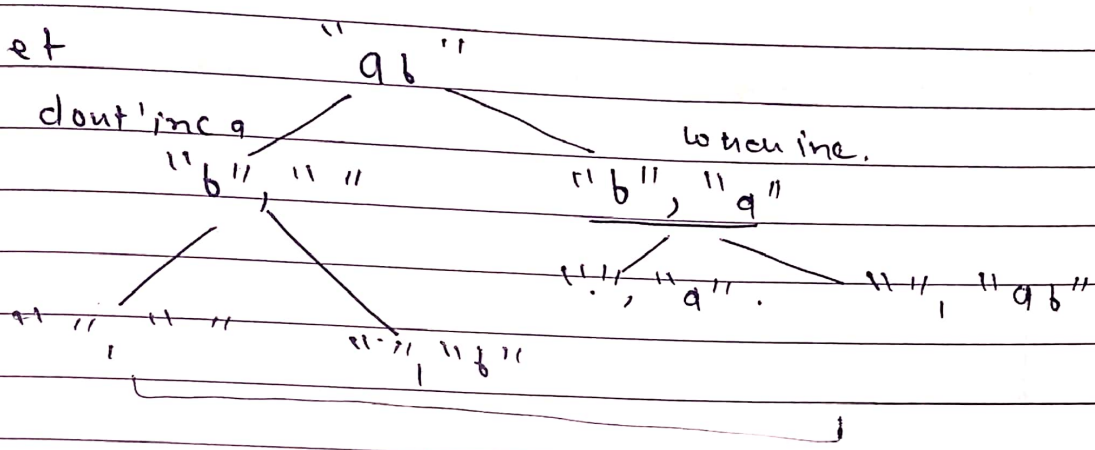
```
#include -----

int keypad(int num, string output[], string dict[])
{
    if(num==0)
    {
        output[0] = "";
        return 1;
    }
    int lastdigit = num%10;
    int smallinput = num/10;
    string smalloutput[10000];

    int smallOutputsize = keypad(smallinput,
                                 smalloutput,
                                 dict);

    string op = dict[lastdigit];
    int k=0;
    for(int i=0; i<op.size(); i++)
    {
        for(int j=0; j<smallOutputsize; j++)
        {
            output[k++] = smalloutput[j] + op[i];
        }
    }
    return k;
}
```

234
4
24

24

```
return k;
}
int keypad (int num, string output[])
{
    string digits [] = { "", "", "abc", "def",
                         "ghi", "jkl", "mno",
                         "tuv", "wxyz" };
    return keypad (num, output, digits
}

int main ()
{
    int num;
    cin >> num;
    string output [10000];
    int count = keypad (num, output);
    for(int i = 0; i < count && i < 10000; i++)
    {
        cout << output [i] << endl;
    }
    return 0;
}
```

How it print :→ String in Rec (



4 dill. outp.

= 4.

```
Void print_subs (string input, string output)
{
    if (input.length() == 0)
    {
        cout << output << endl;
        return;
    }

    print_subs (input.substr(1), output);
print_subs(input.substr(1), output + input[0]);

}

int main()
{
    String input;
    cin >> input;
    string output = "";
    Point. "";
    print_subs (input, output)
}
```

Time Complexity

Selection sort v/s merge sort.

N	Selection sort	merge sort
10	0	2
100	24	6
1000	2352 (0.23 sec)	68
10000	239007 2.	905.
100000	24266045 124.26 sec	13026. → 1485.25 → 0.148 sec
10^6	Hak gaya.	Bhot fast.

Problem to Exp. Analysis

1. Run again
2. Test case (we generally consider worst case)
3. Code each step
4. Time consuming

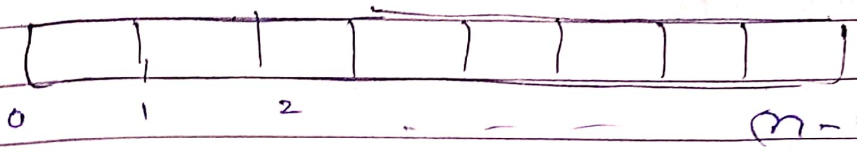
~: Points to Rem ~:

1. no of unit operation. & not time
2. Highest power
3. we don't care
4. don't care abt efficient

Big O Notation

Camlin Page
Date / /

Bubble Sort



1st iteration

$k(n-1)$

$+ k(n-2)$

$+ k(n-3)$

$+ \dots$

$+ k.$

$k(1+2+3+\dots+n-1)$

$\frac{k(n-1)(n)}{2}$

$T(n) = \frac{kn(n-1)}{2}$

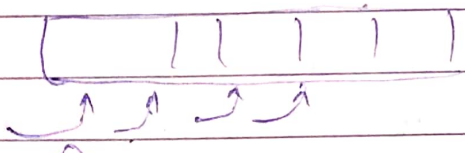
$\frac{kn^2}{2} - \frac{kn}{2}$

$T(n) = \left(\frac{k}{2}\right) n^2$ (c)

$T(\text{BubbleSort}, n) = cn^2$

{Bubblesort is $O(n^2)$}

Linear Search



(x) check whether x present.

if present return (1) else go next.

$k * (1)$ → if element at 0^{th} position (Best case)

Worst case! go to all element

Linear search $O(n)$

Insertion Sort

Compare then swap the insert in sorted array

(eg)

compare ① then if small the swap the

Compare ② then swap if small

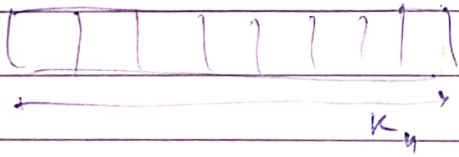
$\therefore k + 2k + 3k + \cdots + (n-1)k$

$k(1+2+3+4+\cdots+(n-1))$

$\frac{k(n)(n-1)}{2} = \frac{kn^2}{2} - \frac{kn}{2}$

2.s in worst case $O(n^2)$

Selection Sort (goto each element & find min)



min = ∞ Int_Max

Compare to & iterate to all element then update min to smallest and place to 1st pos

$k_n + k(n-1) + k(n-2) + \cdots + k$

$k(1+2+3+\cdots+n)$

$k\frac{(n)(n+1)}{2} = \frac{kn^2}{2}$

$O(n^2)$ → Best & Worst case

Recursive Algo

```
int factorial(int n)
{
    if(n==0)
    {
        return 1;
    }
    return n * factorial(n-1);
}
```

$T(n) = k_1 + k_2 + T(n-1)$

Small work of n.

k.

$T(n) = T(n-1) + k$

$T(n-1) = T(n-2) + k$

$T(n-2) = T(n-3) + k$

$T(1) = T(0) + k.$

$T(0) = k.$

$[n+1]$

$T(n) = k + k + k - - - - - -$

$(n+1)$ Times

$T(n) = k(n+1) = kn + k$

$T(n) = kn.$

fact is $O(n)$

Binary Search

go on middle

Boundary find i.e

$T(n) = k + T(n/2)$

go on middle

$T(n/2) = k + T(n/4)$

$T(n/4) = k + T(n/8)$

$T(1) = k.$

$T(n) = k + k + + \ldots$

$n + \frac{n}{2} + \frac{n}{4} + \frac{n}{8} + \ldots \ldots \ldots 1 \left(\frac{n}{2^x}\right)$

$(2^2) \quad (2^3)$

x

1 pai

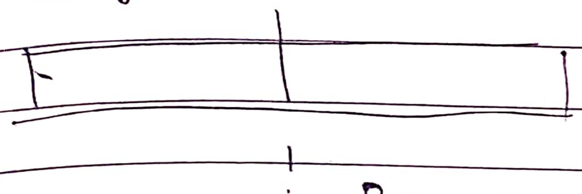
1 pai pohuch

$2^x = n$

$x = \log n$

$O(\log n)$

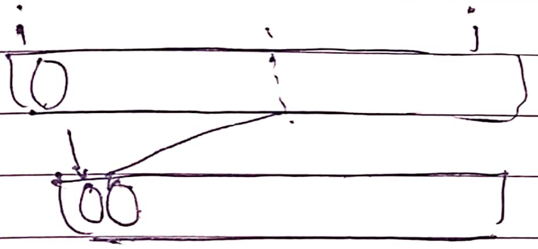
Merge sort



Base case k.

Call on 1/2 half → $T(n/2)$

Call on 2nd half → $T(n/2)$



merg to 1/2's in sec array → $k_1 n$,

copy element from second to input → $k_2 n$

$k_1 n + k_2 n = k_3 n$

$\therefore T(n) = k + 2T(n/2) + k_1 n + k_2 n$

ig.

$T(n) = 2T(n/2) + n(k_1 + k_2)$

$= 2T(n/2) + nk.$

$T(n) = 2T(n/2) + kn. \rightarrow kn$

$2 * [T(n/2) = 2T(n/4) + k(n/2)] \rightarrow kn$

$4 * [T(n/4) = 8T(n/8) + k(n/4)] \rightarrow kn$

$\cdot x$

$\vdots$

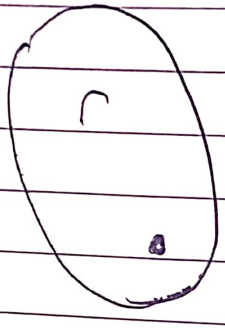
$T(1) = k$ — kn.

$n - \frac{n}{2} \ldots \frac{n}{4} \ldots \frac{n}{2^x}$.

$\frac{n}{2^x} = 1$ $\quad O(n \log n)$

$x = \log n + kn$

Object



Oriented

Programming

Camlin Page
Date / /

Opps :-

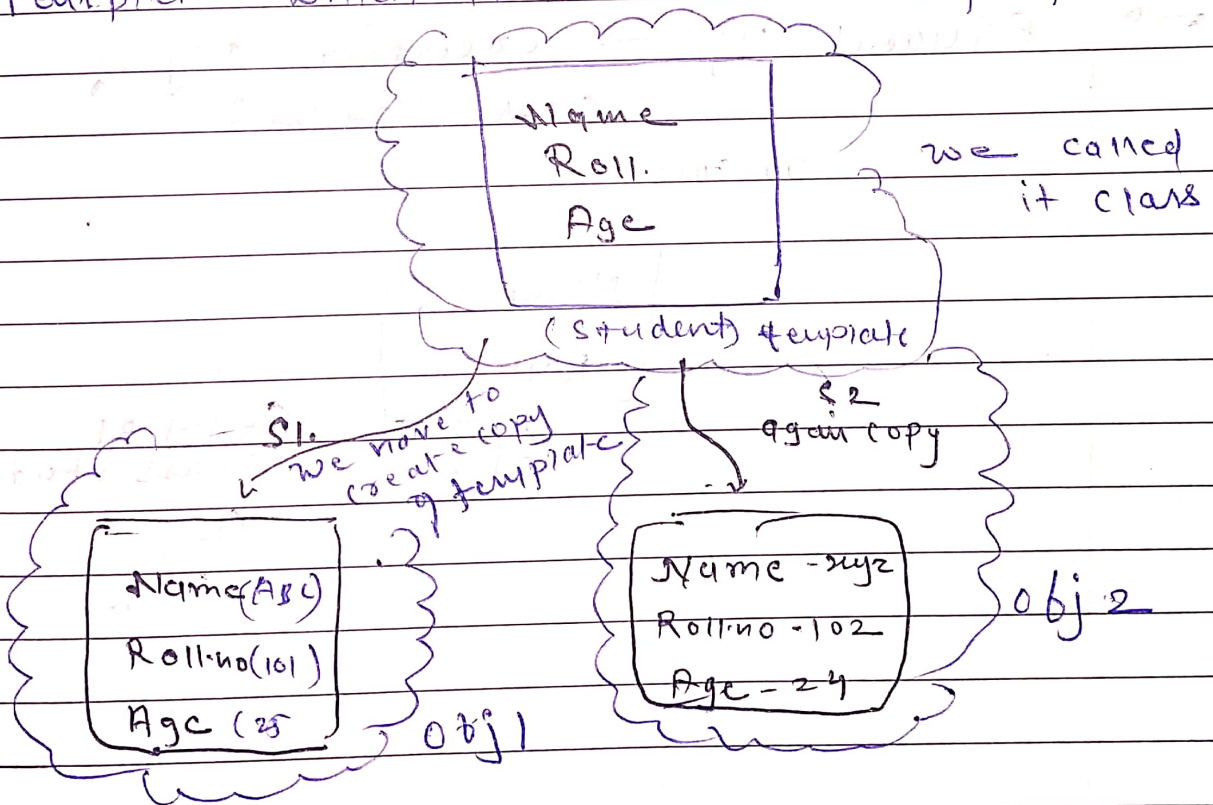
Let student is object it has some prop

- → name
- → Roll.no
- → Age
- → Address
- → contact.no

If we want to perform operation like

Change Address → we have to make
Set Roll.no → f^n

Let we have to take property of 20 student Then we can use obj to and make "Templet" which indicate several properties



Class is a temp that defin Property & f^n of object.

Classes are user defined data type

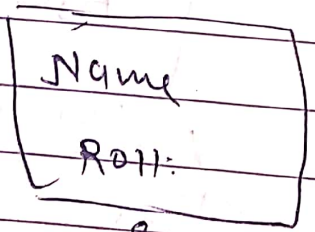
How to made class

Ex1
```
class Student → class name
{
    int rollNumber;
    int age;
};
```

Ex2
```
Class product
{
    int id;
    int weight;
    char name[100];
}
```

→ Student S1; → creating obj of class student

means



S1

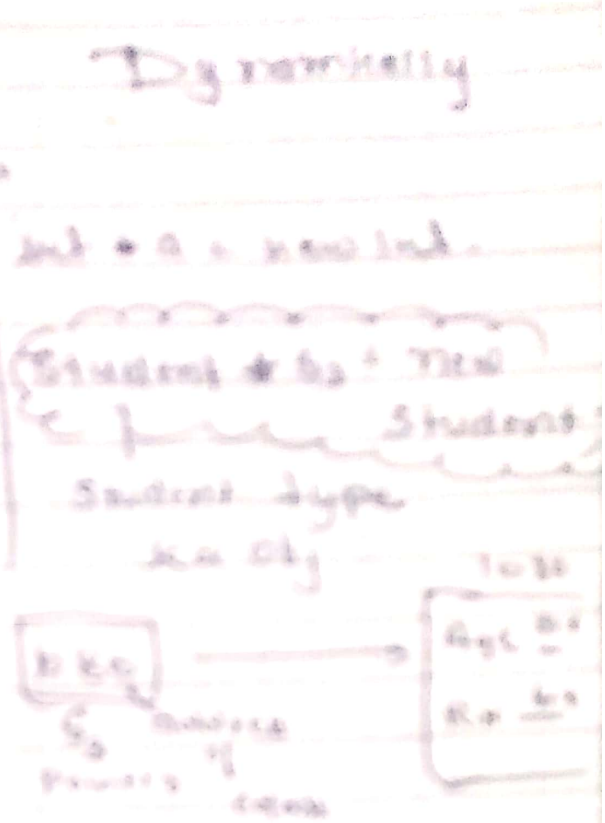
Copy of class student as Sf.

Memory Allocation Techniques

Statically

Student s1;
Student s2;

Dynamically



Access Modifier

- Public
- Private
- Protected

By default Access modifier → Private

We can access the private data member outside the class.

within class we can access public data member

Getters & Setters

How to access private data member of class outside of class?

We can make member fn in public and call fn outside of class.

```
Public:

    int getAge()
    {
        return age;
    }

int main()
{
    cout << "s1 age : " << s1.getAge() <<
    cout << "age " << " " <<
```

Camlin Page
Date / /

Making Getters & setters

```
Class Student {

        Public:
            int rollNumber;
        Private:
            int age;

        Public:
        Void display()
                {
                    cout << age << " " << rollNumber
                                        << endl;
                }
        int getAge()    → getter.
            {
              return age;
            }
        void setAge(int a)   → setter.
            {
              age = a;
            }
};

int main()
        {
            Student S1;
            Student *S2 = new Student;
            S1.setAge(20);
            S2 → setAge(24);
            S1.display();
            S2 → display();
        }
```

Constructors

The fn which has name of the class;

ex → class student{

}

s1 student()

→ default const no Arguments

constructor

→ same name
→ No return type
→ NO input arguments

* we never creats a default constructor when class is made the default const. is ~~auto~~ autometaelly created.

True.

```
student() {

}
```

When we made object constructor(default) is autometically created

```
student s2;
student * s3 = new student;
*s3.student;
s3 → student;
```