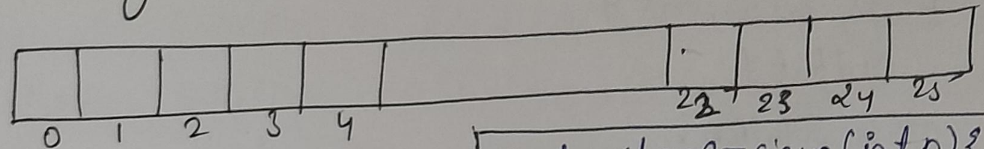
Optimising sieve of ERATOSTHENES



Inner loop

i=2 → 4, 6, 8, 10, 12, 14, 16, 18, 20,

3 → 6, 9, 12, 15, 18, 21, 24

5 → 10, 15, 20, 25

7 → 14, 21, ~~28, 35~~

2x2=4
2x3=6
2x4=8
2x5=10

humme ix2 se shuru nhi karna hai shuru karo ixi se kuki jo ix2 hoga vo pehle se mark hoga agar 5 ki baat karu toh 5x5 pehla hoga jo unmarked hoga uske pehle ka toh unmarked ho chuka hai toh optimization ixi hoga in

```
vector<bool> sieve(int n){
    vector<bool> sieve(n+1, true);
    sieve[0] = sieve[1] = false;

    for(int i=2; i<=n; i++){
        if(sieve[i]==true){
            int j=i*i;        ix2

            while(j<=n){
                sieve[j]=false;
                j+=i;
            }
        }
    }
    return sieve;
}
```

Outer loop Optimization :-

inner loop in j=ixi;

```
while (j<=N){
    sieve[j]=false;

    j+=i;
}
```

if N=25 , i=7 int j= 7x7=49

ye loop nhi chalega →

```
int j=49;
while (j<=25){
} j+=i;
```

Jo outer loop hai vo shure karo i=2 & jao i<=√N

Jo array ban rhi hai 0 se 25 tak agar inner loop ke optim—se √25 ka root niklathe pe 5 milta hai extra work kar rha hai

inner loop bol rha hai ~~ix~~ i=5 , j=ixi= 25 se suru hoga iska ~~matly~~ matlab agar i=5 se jo bhi badi value hai 7 9 11 ye sab jab ayega toh j ki value 25 se bad ~~hai~~ ho jayegi islye outer loop ko utna hi chalo jitne badi array hai

Agar i ki value root n √n se jada lekar chale jati hu toh inner loop kaam nhi karega so lets optimise outer loop as i=2 ; i<=√N outer loop.

(ixi<=n) ←

for(int i=2 ; ixi<=n ; i++)

≃nlog(logn)

Best way to make Sieve

```
vector<bool> sieve(int n){
    vector<bool> sieve (n+1, true);
    sieve[0]= sieve[1] = false;

    for (int i=2; i*i<=n; i++){
        if (sieve[i] == true){

            j = i*i;
            while(j<=n) {
                sieve[j] = false;

                j+=i;
            }
        }
    }
    return sieve;
```

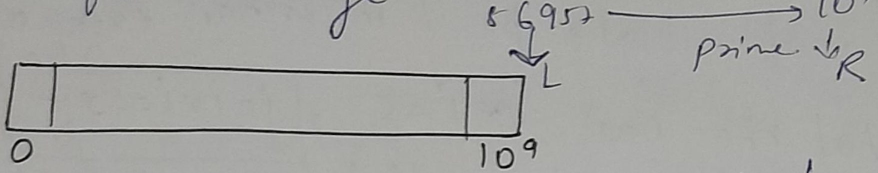
Products
Sum

L, R   Iske beech me jitne bhi prime hai
No btao.

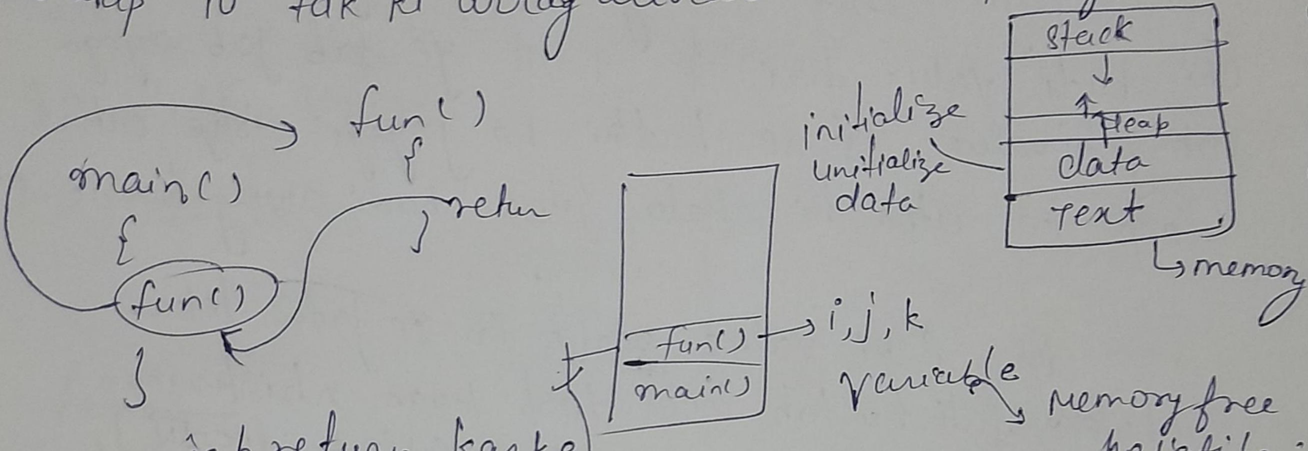
R = 198935

L = 13956

If R is too large   R => 10^9



~~Jab~~ Aap 10^9 tak ki array allocate nhi kar paooge.



jab return karke wapas aa jate ho ye stack se remove ho jata hai.

```
fun()
{
int a[1000];
}
```

iska bhi max size hota hai
Means kissi bhi ~~array~~ func^n me aap array declare kar rhe ho toh uska max^m size → 10^6 ho sakta hai

int, double, char, array, max-size = 10^6
bool array = 10^7
Global array = int, double, char → 10^7
bool → 10^8

source of truth depend computer architecture

* Operating System ki bhi stack ek limit lga ~~de~~ rheta hai

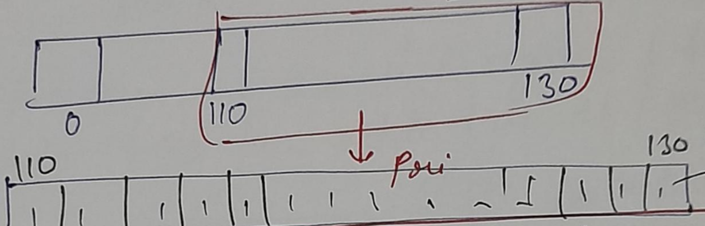
Agar kissi ne bola 10^9 tak ke ~~beech~~ sieve bna do toh that's not possible.

Requirement in Q.

$1 <= (L,R) <= 10^9$ Agar left or Right ki ~~a~~ range di hui hai toh hum (R-L) nikalte hai toh ye 10^6 tak aayega.

$(R-L) <= 10^6$ ← But

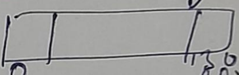
Suppose L = 110, R = 130 $\boxed{(R-L+1) = 21}$ → Max size array ka



isko fill kardugi 0,1 se jha jha 1 hoga vo prime no hoga

Algo

① Generate all prime responsible to mark segmented sieve using $\sqrt{R}$ tak array bnate



Initially all 1.

[yhape vo wale prime nikalne the thai jo Normal sieve me 2 ke multiple, 3 ke multiple nikal rhe the vo marking me help karege]

Using Normal sieve se niklega → $\boxed{N = \sqrt{R}}$ de duga

$N = \sqrt{130}$

$N = 11.4 = 11$

Normal Sieve me N = 1

iske under jitne bhi prime hoge → These prime helps to mark segmented sieve.

age $\boxed{N=11}$ du toh j = 11×11 se suru hota. Inner loop me

② Base Prime = {2, 3, 5, 7, 11}

find first index to start marking

index = 0 → Resemble 110

index 20 → " 130

→ first multiple → L = (10/2)×2

⇒ 110.

Prime ki value 2 hai start

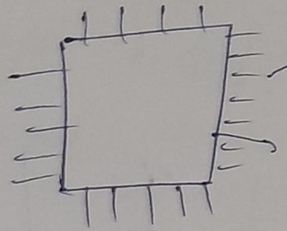
Prime 3 → first multiple ⇒ $(10/3) * 3$ $= (36.6) * 3$ $= 108.$

```
if (firstmul < L){
    first mul += Prime.
}
```

⇒ `int j=(first mul, prime * prime)`

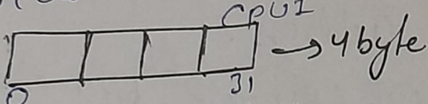
```
vector<bool>sieve = Sieve(
```

Difference b/w 32-bit vs 64-bit OS

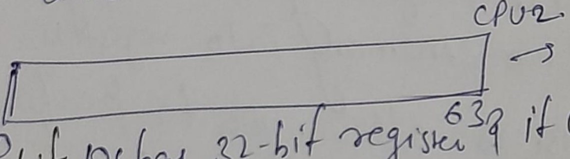


→ billions of transistor
CPU chip motherboard pe embedded hoti hai
→ Transistors hote hai → individual digital circuit.

Register CPU ke andar hote hai jha actual computation hoti hai. It basically holds the address. It is a memory block.



32 bit processor me ~~32-bit~~ jo registers hote hai vo 32-bit ka data hold kar sakte hai ek baar me.

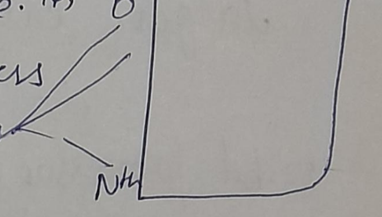


→ Both are diff. architecture.

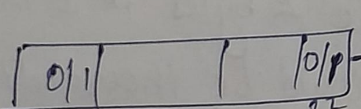
① A 32-bit OS has 32-bit register, it can access 2^{32} unique memory address. i.e. 4GB of physical memory.

② A 64-bit OS has 64-bit registers 2^{64} UMA, i.e. 17,179,869,184 GB of physical address

CPU fetches data from memory address. It can fetch data from 0, from CPU 0+1 also, it can also fetch nth data.

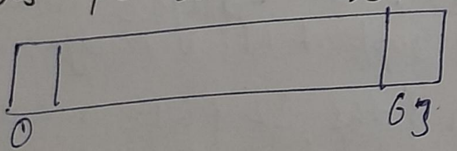


32-bit CPU pe



→ 2^{32} unique address locate kar payega.
4 byte → 4GB RAM ko access kar sakta hai

Registers ko badha dete hai



2^{64} great sol^n ki that we have doubled the size of register. Now we can support greater amount of memory, can allocate more addresses.

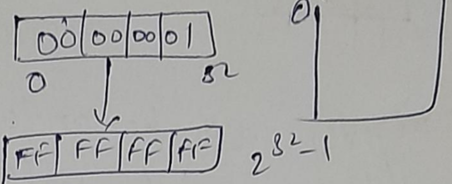
[illegible] ho gya.

2 cycle lag jayege ek 64bit ke addition ko 32 bit me agar karte hai toh CPU ke case me cycle ko sambhalte hai
Jitna kam cycle utna accha

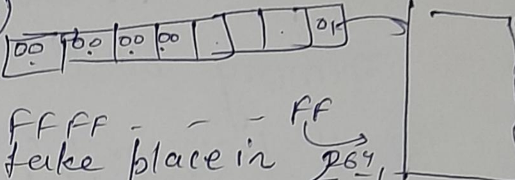
Advantages:-

① Addresseable Space:-

32 bit CPU → 2^{32} bit memory Hexa address.



64 bit CPU → 2^{64} bit memory add. 64



② Performance → All calculations take place in register. When you are performing math in your code, operand are loaded from memory into registers. So, have large registers allow you to perform larger calculation at the same time

32-bit processor can execute 4 bytes of data in 1 instruction cycle while 64-bit means that processor can execute 8 bytes in 1 instruction cycle.

Double amount of work kar paa rha hai 1 cycle me 1sec me

GHz ye hi btata hai cycle.

In 1sec there could be thousands of billion of inst-cycle depending upon a processor design.

③ Resource Usage → 64 bit > 32 bit → agar isme ek extra Ram install kar di toh ye support nhi karega.

④ Comp 64bit CPU run both 32 & 64 → means downward Compatibility hai lekin 32-bit ke andar only 32 bit ka ~~data~~ ka OS chal sakta hai

⑤ Better Graphics Performance - 64 bit > 32 bit
8 bytes graphics Calcu make graphics - intensive apps run faster-