#include <stdio.h>

int main()

{

FILE \*fp = fopen("testdata.in", "r"); //Buka file

int neg\_min=0, pos\_max=0; //Bikin

int num[105];

int size;

while (!feof(fp))

{

fscanf(fp, "%d", "&num[size]");

{

if (num < 0)

{  
 if(arr\_neg>num){  
 arr\_neg = num;

size++;

} // negative

else if (num > 0)

{

arr\_pos[size] = num;

size++;

} // positive

}

}

}

Soal:  
0 1 2 3 4 5 6 7 (index) [i]  
20 17 35 11 -15 -10 -3 -20

size: 8

min\_negative:  
-20, di index 7

max\_positive:   
35, di index 2

Pseudocode:

0 1 2 3 4 5 6 7 (index) [i]  
20 17 35 11 -15 -10 -3 -20

**negative min = 0;   
positive max=0**

**while(not end of file) // 1x doang  
 scan number  
 if(number<0)//negative  
 if(negative min > number)  
 negative min = number  
 if(number>0)//Positive  
 if(positive max < number)  
 positive max = number**

**print positive max - negative min**

**Big-O Notation?**

**O(N) = N → Jumlah elemen = 8**

—--

negative arr = []

neg size = 0

positive arr = []

pos size = 0

while (not EOF)

scan number

if (number < 0) negative arr[neg size] = number

….

for negative arr:

// cari paling kecil

for positive arr:

// cari paling besar

Big-O Notation

O(3N) → O(N)

—--

//Nested for

for () {

for () {

}

}

O(N^2)

Logic:  
buat variable  
negative min dan positive max  
  
kita buat kondisi di **while loop** untuk menchek nomor dan nomor positif  
  
Negatif  
  
Kalau negatif kita cek jika nomor itu lebih kecil daripada nomor 0, kita masukin ke negative min.

Iterasi selanjutnya kita cek jika nomor itu lebih kecil daripada nomor negative min sebelumnya  
  
Ketemu,   
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
2.  
Pseudocode:  
  
struct item {

str nama;  
 var jumlah gudang  
 var jumlah diambil  
 var jumlah total

}

struct item list  
  
function merge(str arr[], int left, int right)  
 if(left<right)

int mid = (left+right)/2;  
 merge(arr, left, mid) //Kiri

merge(arr, mid+1, right) //Kanan  
 mergesort(arr, left, right) //Gabung   
  
function mergesort(str arr[], int left, int right)  
 int mid = (left+right)/2  
 int l1[right-left+1]  
 int r2[right - mid]

int count = 0;   
 str sorted arr[] //temporary array  
 while(l1<=mid && r2 <= right)  
 if(list[l1].nama<list[r2].nama) //Left  
 strcpy(sorted arr[count] = list[r2].name)

count++, r2++

if(list[l1].nama>list[r2].nama) //Right  
 strcpy(sorted arr[count] = list[l1].name)

count++, l1++

// sisa kanan  
 while(r2 <= right){  
 strcpy(sorted arr[count] = list[r2].name)

count++,r2++  
  
 //Sisa kiri  
 while(l1<=mid)  
 strcpy(sorted arr[count] = list[l1].name)

count++, l1++

for(i=0; i < count; i++)  
 list[l1+i] = sorted array temporary

function main {  
 var nomor\_loop = 0;  
 scan nomor\_loop

var index = 0;  
  
 for nomor loop  
 scan list[index].nama

scan list[index].jumlah gudang

scan list[index].jumlah diambil  
 list[index].jumlah total = list[index].jumlah gudang[index] - list[index].jumlah diambil

index++

mergesort(struct list, 0, nomor\_loop-1)

for nomor loop  
 if()

print boots[index]

}

3.  
  
Logic:  
  
  
1. Scanf testcase  
2.   
3.

4.  
5.  
  
  
func binarysearch(int arr[],int left,int right)  
while left > right{  
 int mid = (left+right)/2  
 if(arr[mid] > target){

right = mid - 1

else if(arr[mid] < target)

left = mid+1

else if(arr[mid] == target)

return mid  
}

function main  
 var testcase  
 scan testcase  
 var nomor[105]  
 str nama[105]  
 str namakita, namasepupu  
 var nomorkita, nomorsepupu  
 for testcase  
 scan nomor[index++]  
 for testcase   
 scan nama[index++]  
   
 scan nomorkita, namakita, nomorsepupu, namakita

indexkita = **binarysearch**(nomor, 0, testcase-1, nomorkita)   
 indexsepupu = **binarysearch**(nomor, 0, testcase-1, nomorsepupu)  
  
 if(indexkita==0 & indexsepupu==testcase-1)

printf -1, -1, null, null

else  
 print indexkita+1, indexsepupu+1, nama[indexkita-1], nomor[indexkita+1]

—------------------------------------------------

indexkita = 0

nama[indexkita-1] → Error: Array out of bounds / out of range.

**Edge case:** Jika tidak ada nama orang di depan kamu / di belakang sepupu kamu, maka tampilkan null, dan untuk indexnya tampilkan -1. Index antrian dimulai dari index = 1.