1. You are provided with a set of keywords. For a given input whenever a keyword is encountered, mark it as “Most used”. If the keyword is not present in the given input then, replace it with the data in the input and mark it as “Least used”. When all the keywords are either “Most used” or “Least used”, start replacing the first keyword whenever a new data is encountered in the input.

Explanation:

Keyword set = [2 4 8 9]

When 2 is entered from the keyboard, the keyword 2 is marked as “Most used”. Similarly, when 8 and 9 are entered, they are marked as “Most used”.

When 7 is entered, then from the keyword set 4 is replaced with 7 and is marked as “Least used”. This process continues till all the elements are either marked as “most used” or “Least used” and from then on the replacements starts with the first number.

#include <stdio.h>

void main()

{

int i, n, a[50], k[50],count=0, key,flag=0;

/\* Array a[50] is used to store the elements. Array k[50] is used to find whether the marked as “most used” or “Least used”. If element in array k is set to 1 denotes it is marked, if not it denotes it is unmarked

Count is used to track whether all the element of array are called or not.

Flag is used for setting Most Used and Least Used.

\*/

printf("Enter the size of keyword set.");

scanf("%d",&n);

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

{

scanf("%d",&a[i]);

k[i]=0;

}

while(count != n)

{

printf("Enter a key:");

scanf("%d",&key);

printf("\n");

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

{

if(a[i] == key) // If key is present in the array.

{

k[i]=1;

printf("most used\n");

flag=1;

count++;

break;

}

}

if(flag == 0)

{

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

{

if(k[i] == 0)

{

printf("least used\n");

k[i]=1;

a[i]=key;

count++;

break;

}

}

}

flag=0;

}

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

{

printf("%d\n",a[i]);

}

}

1. In the given array, traverse from left to right replacing the first element with the next greater number present to its right. If no greater number to its right is found, then replace the remaining numbers with -1.

Sample input = [2 4 8 90 77 54 ] Sample output: [4 8 54 -1 -1 -1]

/\* The two loops used in this program is similar to bubbled sorting loop logic.\*/

#include <stdio.h>

void main()

{

int a[50],n,i,j,temp;

printf("Enter the size:");

scanf("%d",&n);

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

scanf("%d",&a[i]);

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

{

temp=a[i];

for(j=i+1;j<n;j++)

{

if(a[j]>a[i])

{

if(temp==a[i])

temp=a[j];

else if(a[j]<temp)

temp=a[j];

}

}

if(temp != a[i])

/\*if temp != a[i] means swapping is done, i.e., larger element is available to the right\*/

{

a[i]=temp;

}

else /\* swapping not done, i.e., larger element is not available to the right\*/

{

a[i]=-1;

i++;

break; /\* Comes out of the loop\*/

}

}

for(;i<n;i++) /\* for all remaining elements value is assigned to -1\*/

a[i]=-1;

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

printf("\n%d",a[i]);

}

3. Divide the given array into two arrays such that both arrays have equal averages

Sample input: [2 4 8 10 16] Sample output: [8] and [2 4 10 16]

Sample input: [2 4 8 10] Sample output: [4 8] and [2 10]

#include <stdio.h>

void main()

{

int a[50],b[50],c[50],n,i,j,l,sum=0,k;

float avg=0;

printf("Enter the total number of elements:");

scanf("%d",&n);

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

{

scanf("%d",&a[i]);

avg += a[i];

}

avg=avg/n;

/\* This loop is used for grouping of elements hence it starts with 1 and goes upto n-1\*/

for(i=1;i<n;i++)

{

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

{

/\*

The below for loop is used for grouping of elements. i.e, when i=1, this loop has sum as a single elment, i=2, this loop takes sum as for 2 elements taken as a pair. And it continues.

\*/

for(j=k;j<k+i;j++)

{

sum=sum+a[j];

}

if(avg == sum/i)

{

printf("\n");

printf("Array 1\n");

for(l=k;l<k+i;l++)

{

printf("%d\t",a[l]);

}

printf("\n");

printf("Array 2\n");

for(l=0;l<k;l++)

{

printf("%d\t",a[l]);

}

for(l=k+i;l<n;l++)

{

printf("%d\t",a[l]);

}

return;

/\*return works same as exit. Since we have found the group, the program is being terminated. Since the return type of main is given void, simply return; is given\*/

}

sum=0;

}

}

/\* The program comes to this region only if the equal average arrays are not possible\*/

printf("\nIt is not possible to split the given array.");

}