Merge Sort

Student : Hussin Almoustafa

Studentnr : 1776495

Merge sort (iteratieve versie) ::

iteratieve merge sort, we doen een bottom-up benadering, dat wil zeggen, beginnen met een array van 2 elementen (we weten dat de array met 1 element al is gesorteerd). Het belangrijkste punt is ook dat, aangezien we niet weten hoe we de array precies moeten verdelen zoals in de top-down-benadering, waarbij de array met 2 elementen de groottereeks 2,1,2,2,1 kan hebben...we in de bottom-up-benadering neem aan dat de array precies is gedeeld door machten van 2 (n/2,n/4….etc) voor een arraygrootte van machten van 2, bijvoorbeeld: n=2,4,8,16.

Dus voor andere invoerformaten zoals 5, 7, 11 zullen we een resterende sublijst hebben die niet de macht van 2 breedte heeft op elk niveau terwijl we blijven samenvoegen en omhoog gaan, deze niet-samengevoegde sublijst die van grootte is die niet is exacte macht van 2, blijft geïsoleerd tot de uiteindelijke samenvoeging.

Het aantal niet-samengevoegde sublijstelementen dat wordt geïsoleerd tot de laatste samenvoegoproep kan worden gevonden met behulp van de rest (n % breedte). De uiteindelijke samenvoeging (wanneer we ongelijke lijsten hebben) kan worden geïdentificeerd door (breedte>n/2).

Aangezien breedte groeit met machten van 2 wanneer breedte == n/2 dan betekent dit dat de invoer al een grootte had in machten van 2, anders als breedte < n/2 dan hebben we de uiteindelijke samenvoeging nog niet bereikt, dus wanneer breedte > n /2 we moeten een niet-samengevoegde oneven lijst in behandeling hebben, daarom resetten we alleen in dat geval mid.

Code :

```
#include<iostream>

using namespace std;



void merge(int a[], int l, int m, int r){

    int temp[m-l +1], temp2[r-m];

    for(int i=0; i<(m-l + 1); i++)
        temp[i] = a[l+i];

    for(int i=0; i<(r-m); i++)
        temp2[i] = a[m+1+i];

    int i=0;
    int j=0;
    int k=l;


    while (i<(m-l+1) && j<(r-m))
    {
        if(temp[i]<temp2[j])
            a[k++]= temp[i++];

        else
            a[k++]= temp2[j++];

    }
```

```
    while (i<(m-l+1))
    {
        a[k++]= temp[i++];
    }

    while (j<(r-m))
    {
        a[k++]=temp2[j++];
    }
}

void merge_sort(int a[], int l, int r){

    if(l< r){
        int m = (l+r)/2;
        merge_sort(a,l,m);
        merge_sort(a,m+1,r);
        merge(a,l,m,r);
    }
}


void print(int a[], int n)
{
    for(int i=0; i<n; i++)
        cout<<a[i]<<" ";
    cout<<endl;
}
int main()
{
    int numToSort = 0;
    int index;
```

```
}
int main()
{
    int numToSort = 0;
    int index;

    int * gArray = new int[numToSort];

    cout << "Welkom bij het Merge Sort-programma!" << endl;
    cout << "Voer a.u.b. in hoeveel getallen worden gesorteerd: ";
    cin >> numToSort;
    cout << endl;

     //gebruiker voert elementen van de array in
    for (index = 0; index < numToSort; index++)
    {

        cout << "Voer alstublieft een nummer in: ";
        cin >> gArray[index];
    cout << endl;
    }
    cout<<"Unsorted array; ";
    print(gArray,numToSort);
    merge_sort(gArray,0,numToSort-1);
    cout<<"\nSorted array: ";
    print(gArray,numToSort);
    return 0;
}
```