Array using C++ modified

```cpp
#include <iostream>

using namespace std;
class Array
{
private:
    int *A;
    int size;
    int length;
    void swap(int *x,int *y);

public:
    Array()
    {
        size=10;
        length=0;
        A=new int[size];
    }
    Array(int sz)
    {
        size=sz;
        length=0;
        A=new int[size];
    }
    ~Array()
    {
        delete []A;
    }
    void Display();
    void Append(int x);
    void Insert(int index,int x);
    int Delete(int index);
    int LinearSearch(int key);
    int BinarySearch(int key);
   int Get(int index);
    void Set(int index,int x);
    int Max();
    int Min();
    int Sum();
    float Avg();
    void Reverse();
    void Reverse2();
    void InsertSort(int x);
    int isSorted();
    void Rearrange();
    Array* Merge(Array arr2);
    Array* Union(Array arr2);
    Array* Diff(Array arr2);
    Array* Inter(Array arr2);
};

void Array::Display()
{
    int i;
    cout<<"\nElements are\n";
    for(i=0;i<length;i++)
        cout<<A[i]<<" ";
}

void Array::Append(int x)
{
    if(length<size)
        A[length++]=x;

}

void Array::Insert(int index,int x)
{
    int i;
    if(index>=0 && index <=length)
    {
        for(i=length;i>index;i--)
            A[i]=A[i-1];
        A[index]=x;
        length++;
    }
}
int Array::Delete(int index)
{
    int x=0;
    int i;

    if(index>=0 && index<length)
    {
        x=A[index];
        for(i=index;i<length-1;i++)
            A[i]=A[i+1];
        length--;
        return x;
    }

    return 0;
}
void Array::swap(int *x,int *y)
{
    int temp;
    temp=*x;
    *x=*y;
    *y=temp;
}
int Array::LinearSearch(int key)
{
    int i;
    for(i=0;i<length;i++)
    {
        if(key==A[i])
        {
            swap(&A[i],&A[0]);
            return i;
        }
    }
    return -1;
```

```
}
int Array::BinarySearch(int key)
{
    int l,mid,h;
    l=0;
    h=length-1;

    while(l<=h)
    {
        mid=(l+h)/2;
        if(key==A[mid])
            return mid;
        else if(key<A[mid])
            h=mid-1;
        else
            l=mid+1;
    }
    return -1;
}

int Array::Get(int index)
{
    if(index>=0 && index<length)
        return A[index];
    return -1;
}
void Array::Set(int index,int x)
{
    if(index>=0 && index< length)
        A[index]=x;
}
int Array::Max()
{
    int max=A[0];
    int i;
    for(i=1;i<length;i++)
    {
        if(A[i]>max)
            max=A[i];
    }
    return max;
}
int Array::Min()
{
    int min=A[0];
    int i;
    for(i=1;i<length;i++)
    {
        if(A[i]<min)
            min=A[i];
    }
    return min;
}

int Array::Sum()
{
    int s=0;
    int i;
    for(i=0;i<length;i++)
        s+=A[i];

    return s;
}

float Array::Avg()
{
    return (float)Sum()/length;
}

void Array::Reverse()
{
    int *B;
    int i,j;

    B=(int *)malloc(length*sizeof(int));
    for(i=length-1,j=0;i>=0;i--,j++)
        B[j]=A[i];
    for(i=0;i<length;i++)
        A[i]=B[i];

}
void Array::Reverse2()
{
    int i,j;
    for(i=0,j= length-1;i<j;i++,j--)
    {
        swap(& A[i],& A[j]);
    }
}
void Array::InsertSort(int x)
{
    int i= length-1;
    if( length== size)
        return;
    while(i>=0 &&  A[i]>x)
    {
        A[i+1]= A[i];
        i--;
    }
    A[i+1]=x;
    length++;

}
int Array::isSorted()
{
    int i;
    for(i=0;i<length-1;i++)
    {
        if(A[i]>A[i+1])
            return 0;
    }
    return 1;
}
```

```
void Array::Rearrange()
{
    int i,j;
    i=0;
    j= length-1;

    while(i<j)
    {
        while( A[i]<0)i++;
        while( A[j]>=0)j--;
        if(i<j)swap(& A[i],& A[j]);
    }

}

Array* Array::Merge(Array arr2)
{
    int i,j,k;
    i=j=k=0;

    Array *arr3=new Array(length+arr2.length);

    while(i<length && j<arr2.length)
    {
        if(A[i]<arr2.A[j])
            arr3->A[k++]=A[i++];
        else
            arr3->A[k++]=arr2.A[j++];
    }
    for(;i<length;i++)
        arr3->A[k++]=A[i];
    for(;j<arr2.length;j++)
        arr3->A[k++]=arr2.A[j];
    arr3->length=length+arr2.length;

    return arr3;
}

Array* Array::Union(Array arr2)
```

```
{
    int i,j,k;
    i=j=k=0;

    Array *arr3=new Array(length+arr2.length);

    while(i<length && j<arr2.length)
    {
        if(A[i]<arr2.A[j])
            arr3->A[k++]=A[i++];
        else if(arr2.A[j]<A[i])
            arr3->A[k++]=arr2.A[j++];
        else
        {
            arr3->A[k++]=A[i++];
            j++;
        }
    }
    for(;i<length;i++)
        arr3->A[k++]=A[i];
    for(;j<arr2.length;j++)
        arr3->A[k++]=arr2.A[j];

    arr3->length=k;

    return arr3;
}

Array* Array::Inter(Array arr2)
{
    int i,j,k;
    i=j=k=0;

    Array *arr3=new Array(length+arr2.length);

    while(i<length && j<arr2.length)
    {
        if(A[i]<arr2.A[j])
```

```
                i++;
            else if(arr2.A[j]<A[i])
                j++;
            else if(A[i]==arr2.A[j])
            {
                arr3->A[k++]=A[i++];
                j++;
            }
        }

        arr3->length=k;

        return arr3;
}

Array* Array::Diff(Array arr2)
{
        int i,j,k;
        i=j=k=0;

        Array *arr3=new Array(length+arr2.length);

        while(i<length && j<arr2.length)
        {
            if(A[i]<arr2.A[j])
                arr3->A[k++]=A[i++];
            else if(arr2.A[j]<A[i])
                j++;
            else
            {
                i++;
                j++;
            }
        }
        for(;i<length;i++)
            arr3->A[k++]=A[i];
```

```
    arr3->length=k;

    return arr3;
}

int main()
{
    Array *arr1;
    int ch,sz;
    int x,index;

    cout<<"Enter Size of Array";
    scanf("%d",&sz);
    arr1=new Array(sz);

    do
    {
        cout<<"\n\nMenu\n";
        cout<<"1. Insert\n";
        cout<<"2. Delete\n";
        cout<<"3. Search\n";
        cout<<"4. Sum\n";
        cout<<"5. Display\n";
        cout<<"6.Exit\n";

        cout<<"enter you choice ";
        cin>>ch;

        switch(ch)
        {
            case 1: cout<<"Enter an element and
index ";
                cin>>x>>index;
                arr1->Insert(index,x);
                break;
            case 2: cout<<"Enter index ";
                cin>>index;
                x=arr1->Delete(index);
```

```
                    cout<<"Deleted Element is"<<x;
                    break;
                case 3:cout<<"Enter element to search
";
                    cin>>x;
                    index=arr1->LinearSearch(x);
                    cout<<"Element index "<<index;
                    break;
                case 4:cout<<"Sum is "<<arr1->Sum();
                    break;
                case 5:arr1->Display();

        }
    }while(ch<6);
    return 0;
}
```