Leet code problems

1)Maximum product subarray(152)

solution:

```
int maxProduct(vector<int>& nums) {
        if (nums.size()==1)
        {
            return nums[0];
        }
        int maxsofar=nums[0];
        int minsofar=nums[0];
        int finalmax=nums[0];
        int prev;
        for(int i =1;i<nums.size();i++)
        {
          prev=maxsofar;
            maxsofar=max(nums[i],max(nums[i]*maxsofar,nums[i]*minsofar));
            minsofar=min(nums[i],min(nums[i]*minsofar,nums[i]*prev));
            finalmax=max(maxsofar,finalmax);
        }
        return finalmax;
    }
```

2)subarray product less than k(713)

Solution:

```
int numSubarrayProductLessThanK(vector<int>& nums, int k) {
        int l=0,r=0;
        int result=0;
        int product=1;
        while(r<nums.size())
        {
            product*=nums[r];
            while(product>=k&&l<=r)
            {
                product/=nums[l];
                l++;
            }
            result+=r-l+1;
            r++;

        }
        return result;
    }
```

3 subarray sum equal to k

Solution

```
int subarraySum(vector<int>& nums, int k) {
           int i ,j,sums=0,p=0;
          for( i =0;i<nums.size();i++)
          {
               sums=nums[i];
              if(nums[i]==k)
              {
                  p+=1;
              }

              for( j=i+1;j<nums.size();j++)
```

```
            {
                sums+=nums[j];

                if(sums==k)
                {
                    p+=1;
                }

            }
        }

        return p;
    }
```

4) maximum product of three numbers(628)

Solutions

```
int maximumProduct(vector<int>& nums) {
        int c;
        sort(nums.begin(),nums.end());
        c=nums.size();
        return max(nums[c-1]*nums[c-2]*nums[c-3],nums[0]*nums[1]*nums[c-1]);

    }
```