LEETCODE 1 - MINSTACK

```
typedef struct {
    int size;
    int top;
    int *s;
    int * minstack;
} Minstack;

Minstack * minStackCreate ()
{
    MinStack *st = (Minstack *) malloc (sizeof(
                                    MinStack));
    if(st == NULL)
    {   printf ("Memory allocation failed ");
        exit(0);
    }
    st -> size = 5000000;
    st -> top = -1;
    st -> s = (int *) malloc (st -> size * sizeof(int));
    st -> minstack = (int *) malloc (st -> size *
                                sizeof (int));
    if(st -> s == NULL)
    {
        printf ("Memory allocation failed");
        free(st -> s);
        free (st -> minstack);
    }
    return st;
}
```

```
void minstackPush (MinStack * obj, int val)
{
    if (obj->top == obj->size-1)
        printf("stack overflow");
    else {
      obj->top++;
      obj->s[obj->top] = val;
      if (obj->top==0 || val < obj->minstack[obj->
                                    top-1])
      {  obj->minstack[obj->top] = val;
      }
      else {
        obj->minstack[obj->top] = obj->minstack
                                  [obj->top-1];
      }
    }
}

void minStackPop (MinStack * obj)
{
    int value;
    if (obj->top == -1)
        printf("Underflow");
    else
    {
        value = obj->s[obj->top];
        obj->top--;
        printf("%d is popped \n", value);
    }
}
```

```
int minstackTop (Minstack* obj)
{
    int value = -1;
    if (obj -> top == -1)
        printf ("Underflow");
    else
    {   value = obj -> s[obj -> top];
        return value;
    }
}

int minStackFree (Minstack * obj)
{
    free (obj -> s);
    free (obj -> minstack);
    free (obj);
}
```