Session Three

> *front/jeason.h*

```
#include <malloc/malloc.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define maxsize 10

//
typedef struct stack {
  int num[maxsize];
  int top;
} sqstack, *sqslink;
```

> *q1.c*

```
#include "../front/jeason.h"

int Push(sqslink s, int tag) {
  if (s->top >= maxsize - 1)
    return 0;
  else {
    s->top++;
```

```
    s->num[s->top] = tag;
    return 1;
  }
}

int Pop(sqslink s) {
  if (s->top < 0)
    return 0;
  else {
    s->top--;
    return s->num[s->top + 1];
  }
}

int Getstop(sqslink s) {
  if (s->top < 0)
    return 0;
  else {
    return s->num[s->top];
  }
}

void PrintStack(sqslink s) {
  for (int i = 0; i < maxsize; i++) {
      printf("%d ", Getstop(s));
  }
}
```

```
int main(void) {
  sqslink newStack = (sqslink)malloc(sizeof(sqstack));
  for (int i = 1; i <= 5; i++) {
    Push(newStack, i);
  }
  PrintStack(newStack);
}
```

> *Result Y: [1,2,3,4,5],[1,2,3,5,4],[1,3,2,4,5],[1,2,4,3,5],[2,1,3,4,5]*

> *Result N: [5,3,2,1,4],[5,2,1,4,3],[5,1,2,3,4],[4,2,1,3,5],[4,1,2,3,5]*

> *q2.c*

```
#include "../front/jeason.h"

int Push(sqslink s, int tag) {
  if (s->top >= maxsize - 1)
    return 0;
  else {
    s->top++;
    s->num[s->top] = tag;
    return 1;
  }
}

int Pop(sqslink s) {
```

```
  if (s->top < 0)
    return 0;
  else {
    s->top--;
    return s->num[s->top + 1];
  }
}

int Getstop(sqslink s) {
  if (s->top < 0)
    return 0;
  else {
    return s->num[s->top];
  }
}

void PrintStack(sqslink s) {
  for (int i = 0; i < maxsize; i++) {
    printf("%d ", Getstop(s));
  }
}

int main(void) {
  printf("%d", 5 / 2);
  sqslink newStack = (sqslink)malloc(sizeof(sqstack));
  int DATA[maxsize], n;
  printf("Please input arr length\n");
  scanf("%d", &n);
```

```
  printf("Please input int arr\n");
  for (int i = 0; i < n; i++) {
    scanf("%d", (DATA + i));
  }
  for (int j = 0; j < n; j++) {
    if ((n % 2 == 0 && j <= n / 2) || (n % 2 != 0 && j <=
 n / 2)) {
      Push(newStack, *(DATA + j));
    } else if ((n % 2 == 0 && j > n / 2) || (n % 2 != 0 &
& j > ((n/2)+1))) {
      if (DATA[j] != Getstop(newStack)) {
        printf("Nope!!");
      }
    }
  }
  printf("Yepe!!");
}
```