runner% **cat lists.c**

```
/* lists: test recursive printing of linked lists */
#include <stdio.h>
#include "stack.h"

void main(void)
{
   char c;
   while ((c = getchar()) != 'q') {
      switch (c) {
      case 'i': c = getchar();
         if (push(c)) printf("No room\n");
         else printf("%c pushed\n", c);
         break;
      case 'r': if (pop(&c))
            printf("Empty stack\n");
         else printf("%c popped\n", c);
         break;
      case 'p': c = getchar();
         if (c == 's') printstack();
         else if (c == 'r')
            printstackrecurs();
         else if (c == 'b')
            printstackreverse();
         else printf("Input error\n");
         printf("\n");
         break;
         default: printf("Input error\n");
      }
      while ((c = getchar()) != '\n')
         ;
    }
}
/*-------------------------------------------------------*/
```

runner% **cat stack.h**

```
/* stack.h: stack header file */
typedef char Stacktype;
int pop(Stacktype *); /* pop an item.
         Return 0 if ok, -1 if not */
int push(Stacktype); /* push an item.
         Return 0 if ok, -1 if not */
int empty(void);
int full(void);
void printstack(void);
void printstackrecurs(void);
void printstackreverse(void);
/*-------------------------------------------------------*/
```

runner% **cat stack.c**

```
/* stack.c: stack implementation */
#include <stdio.h>
#include <stdlib.h>
#include "stack.h"
struct snode {
   Stacktype c;
   struct snode *next;
};
static void printstack2(struct snode *);
static void printstackrecurs2(struct snode *);
static void printstackreverse2(struct snode *);
static struct snode *s = NULL;
static int filledup = 0;

int pop(Stacktype *ch)
{
   struct snode *ssave;
   if (empty()) return -1;
   *ch = s -> c;
   ssave = s;
   s = s -> next;
   free(ssave);
   return 0;
}

int push(Stacktype ch)
{
   struct snode *t = (struct snode *)
      malloc(sizeof(struct snode));
   if (t == NULL) {
      filledup = 1;
      return -1;
   }
   t -> next = s;
   t -> c = ch;
   s = t;
   return 0;
}

int empty(void)
{
   return s == NULL;
}
int full(void)
{
   return filledup;
}
```

```
void printstack(void)
{
   printstack2(s);
}

static void printstack2(struct snode *s)
{
   while (s != NULL) {
      printf("%c", s -> c);
      s = s -> next;
   }
}

void printstackrecurs(void)
{
   printstackrecurs2(s);
}

static void printstackrecurs2(struct snode *s)
{
   if (s != NULL) {
      printf("%c", s -> c);
      printstackrecurs2(s -> next);
   }
}

void printstackreverse(void)
{
   printstackreverse2(s);
}

static void printstackreverse2(struct snode *s)
{
   if (s != NULL) {
      printstackreverse2(s -> next);
      printf("%c", s -> c);
   }
}
```

```
runner% cc -o lists lists.c stack.c
runner% lists
ia
a pushed
ib
b pushed
ic
c pushed

...

iw
w pushed
ix
x pushed
iy
y pushed
iz
z pushed
ps
zyxwvutsrqponmlkjihgfedcba
pr
zyxwvutsrqponmlkjihgfedcba
pb
abcdefghijklmnopqrstuvwxyz
q
runner% lists
ia
a pushed
ib
b pushed
ic
c pushed
pr
cba
pb
abc
r
c popped
r
b popped
r
a popped
q
runner%
```