HW2

(1)

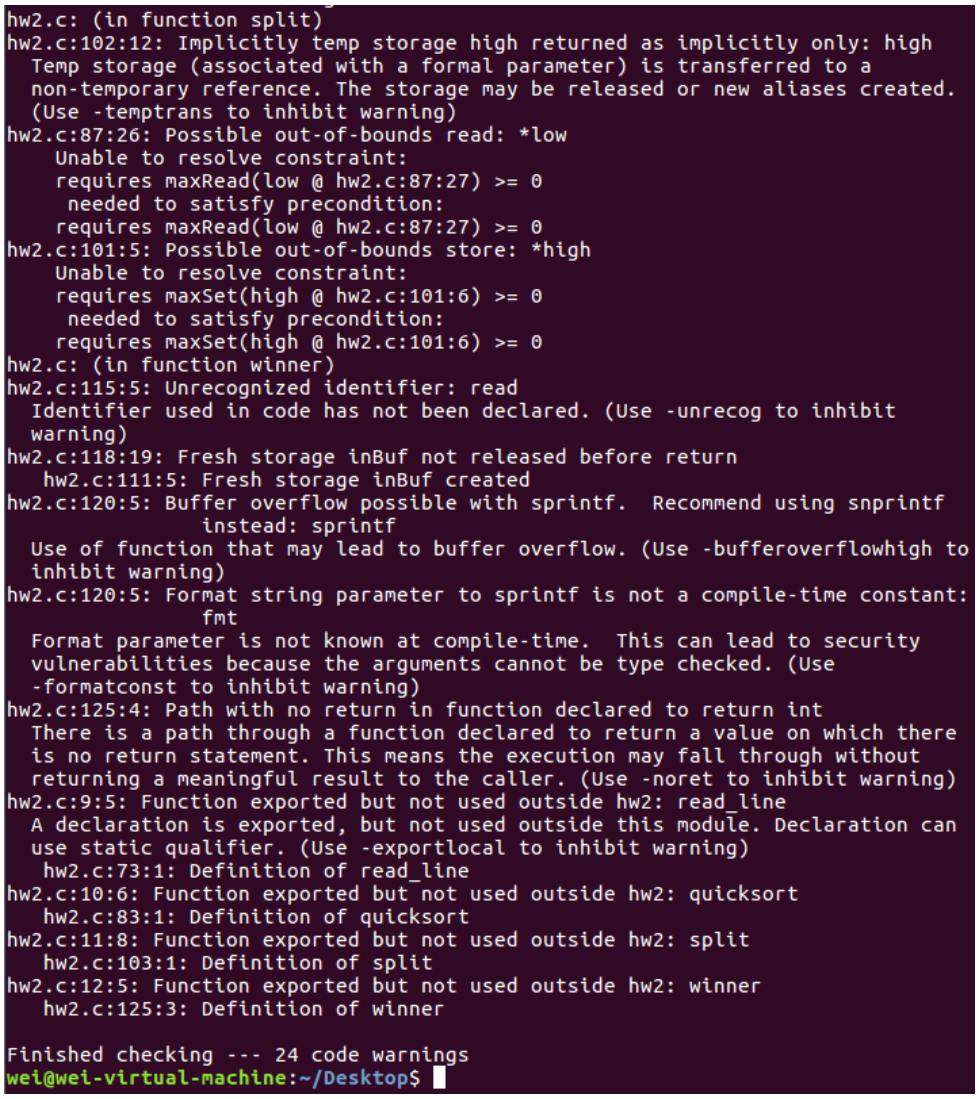
Kiuwan

	Files	Defects	Rule	Priority	Characteristic	Language	Effort
Σ		36					14h 15
▸	1	2	CERT C EXP33: Do not reference uninitialized memory		Security	C	12m
▸	1	1	MISRA 16.2: Functions shall not call themselves, either directly or indirectly		Reliability	C	4h 00
▸	1	12	CERT C FIO33: Detect and handle input/output errors resulting in undefined behavior		Security	C	6h 00
▸	1	3	MISRA 12.13: The increment (++) and decrement (--) operators shall not be mixed with other operators in an expression		Reliability	C	18m
▸	1	2	MISRA 12.5: The operands of a logical && or || shall be primary-expressions		Reliability	C	12m
▸	1	1	MISRA 16.5: Functions with no parameters shall be declared with parameter type void		Maintainability	C	03m
▸	1	1	MISRA 13.1: Assignment operators shall not be used in expressions that yield a boolean value		Reliability	C	06m
▸	1	5	MISRA 14.9: If-else statements must use braces		Reliability	C	30m
▸	1	4	MISRA 14.8: Loops must use braces to delimit loop body		Reliability	C	24m
▸	1	2	Only one 'return' statement per function		Reliability	C	1h 00
▸	1	3	MISRA 14.6: For any iteration statement there shall be at most one break statement used for loop termination		Maintainability	C	1h 30

Splint

```
wei@wei-virtual-machine:~/Desktop$ splint hw2.c +bounds -paramuse -varuse
Splint 3.1.2 --- 03 May 2009

hw2.c: (in function main)
hw2.c:20:5: Return value (type int) ignored: scanf("%d", &cho)
  Result returned by function call is not used. If this is intended, can cast
  result to (void) to eliminate message. (Use -retvalint to inhibit warning)
hw2.c:22:9: Return value (type int) ignored: getchar()
hw2.c:25:9: Return value (type int) ignored: winner()
hw2.c:40:19: Passed storage word not completely defined (*word is undefined):
                 read_line (word, ...)
  Storage derivable from a parameter, return value or global is not defined.
  Use /*@out@*/ to denote passed or returned storage which need not be defined.
  (Use -compdef to inhibit warning)
hw2.c:40:9: Return value (type int) ignored: read_line(word, 20)
hw2.c:54:16: Passed storage words not completely defined (*words is undefined):
                 quicksort (words, ...)
hw2.c:54:23: Passed storage words not completely defined (*words is undefined):
                 quicksort (..., words + num_words - 1)
hw2.c:58:23: Possible out-of-bounds read: words[i]
    Unable to resolve constraint:
    requires maxRead(words @ hw2.c:58:23) >= i @ hw2.c:58:29
     needed to satisfy precondition:
    requires maxRead(words @ hw2.c:58:23) >= i @ hw2.c:58:29
  A memory read references memory beyond the allocated storage. (Use
  -boundsread to inhibit warning)
hw2.c: (in function read_line)
hw2.c:68:11: Operands of != have incompatible types (int, char):
                 (ch = getchar()) != '\n'
  A character constant is used as an int. Use +charintliteral to allow
  character constants to be used as ints.  (This is safe since the actual type
  of a char constant is int.)
hw2.c:70:13: Assignment of int to char: str[i++] = ch
  To make char and int types equivalent, use +charint.
hw2.c:71:5: Possible out-of-bounds store: str[i]
    Unable to resolve constraint:
    requires maxSet(str @ hw2.c:71:5) >= i @ hw2.c:71:9
     needed to satisfy precondition:
    requires maxSet(str @ hw2.c:71:5) >= i @ hw2.c:71:9
  A memory write may write to an address beyond the allocated buffer. (Use
  -boundswrite to inhibit warning)
hw2.c: (in function quicksort)
hw2.c:83:2: Fresh storage middle not released before return
  A memory leak has been detected. Storage allocated locally is not released
  before the last reference to it is lost. (Use -mustfreefresh to inhibit
  warning)
   hw2.c:80:5: Fresh storage middle created
```



沒有用到 winnner()的回傳值，可以宣告為 void

沒有用到 read_line()的回傳值，可以宣告為 void

在 quicksort()結束後沒有把 middle free 掉

在 winner()裡的 read 沒有宣告

在 winner()因為 outBuf=NULL 結束後沒有把 inBuf free 掉

在 winner()裡的 sprintf 可能會造成 buffer overflow，可以用 snprintf

在 winner()裡的 sprintf 的參數 fmt，在編譯時無法辨識

下面 winner()的定義裡的參數要加 void

if-else、for、while 最好用括號

```c
// modified
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_WORDS 50
#define WORD_LEN 20
#define BUF_SIZE (1024)

void read_line(char str[], int n);
void quicksort(char **low, char **high);
char **split(char **low, char **high);
void winner(void);

int main(void)
{
    char *words[MAX_WORDS], word[WORD_LEN+1];
    int i, cho, num_words = 0;

    printf("Enter Choice: 1:sort or 2:game -> ");
    scanf("%d", &cho);
    if(cho==1){
        getchar();
    }
    else if(cho==2) {
        winner();
        return 0;
    }
    else {
        printf("Error!!");
        return 0;
    }

    for (;;) {
        if(num_words == MAX_WORDS) {
            printf("-- No space left --\n");
            break;
        }

        printf("Enter word: ");
        read_line(word, WORD_LEN);
        if(strlen(word) == 0) {
            break;
        }

        words[num_words] = malloc(strlen(word) + 1);
        if(words[num_words] == NULL) {
            printf("-- No space left --\n");
            break;
        }

        strcpy(words[num_words], word);
        num_words++;
    }
```

```
    }

    quicksort(words, words + num_words - 1);

    printf("\nIn sorted order:");
    for(i = 0; i < num_words; i++) {
        printf(" %s", words[i]);
    }
    printf("\n");

    return 0;
}

void read_line(char str[], int n)
{
    int ch, i = 0;

    while((ch = getchar())!='\n') {
        if (i<n) {
            str[i++] = ch;
        }
    }
    str[i]='\0';
}

void quicksort (char **low, char **high)
{
    char **middle;

    if(low >= high) {
        return;
    }
    middle = split(low, high);
    quicksort(low, middle - 1);
    quicksort(middle + 1, high);
    free(middle);
}
```

```
}

char **split(char **low, char **high)
{
    char *part_element = *low;

    for(;;){
        while(low<high&&strcmp(part_element, *high)<=0) {
            high--;
        }
        if(low>=high) {
            break;
        }
        *low++ = *high;

        while(low<high&&strcmp(*low, part_element)<=0) {
            low++;
        }
        if(low>=high) {
            break;
        }
        *high-- = *low;
    }

    *high=part_element;
    return high;
}

void winner(void) {
    char* inBuf;
    char* outBuf;
    char* fmt = "the winner is: %s";

    printf("Enter name:\n");
    inBuf=(char*)malloc(BUF_SIZE);
    if(inBuf==NULL){
        return;
    }
    getchar();
    read_line(inBuf, BUF_SIZE);
    outBuf=(char*) malloc(BUF_SIZE);
    if(outBuf==NULL){
        return;
    }
    snprintf(outBuf, BUF_SIZE, fmt, inBuf);
    fprintf(stdout,"%s\n", outBuf);
    fprintf(stderr,"%s\n", outBuf);
    free(inBuf);
    free(outBuf);
}
```

(2)

之前寫的作業

```c
#include<stdio.h>
#include<stdlib.h>

#define L 20

typedef struct node1{
    char data;
    struct node1 *next;
}NODE1;

typedef struct node2{
    char data;
    struct node2 *next;
    struct node2 *prev;
}NODE2;

void push(NODE1**, char);
void pop(NODE1**);
void in(NODE1**, char, int);
void de(NODE1**, char);
int op(NODE1*);

void enquence(NODE2**, NODE2**, char);
void dequence(NODE2**, NODE2**);
void in2(NODE2**, NODE2**, char, int);
void de2(NODE2**, NODE2**, char);
int op2(NODE2*);

int main(){
    char voc = 'A';
    char str[L] = {'\0'};
    int i, j, n, c, cc;
    NODE1 *top = NULL, *tmp = NULL;
    NODE2 *tail = NULL, *head = NULL, *tmp2 = NULL;

    for(i = 0; i < 4; i++){
        push(&top, voc);
        voc++;
    }
    voc = 'A';
    for(i = 0; i < 4; i++){
        enquence(&head, &tail, voc);
        voc++;
    }
    printf("Please enter sequence: ");
```

```c
    printf("Please enter sequence: ");
    while(1){
        for(j = 0; j < L; j++){
            scanf("%c", &str[j]);
            if(str[j] == '\n'){
                str[j] = '\0';
                break;
            }
        }
        i = 0;
        if((str[i] == '-') && (str[i+1] == '1'))
            break;
        else if(str[i] == 'S'){
            for(j = 0; j < L; j++)
                str[j] = '\0';
            printf("\nCommand: ");
            while(1){
                for(j = 0; j < L; j++){
                    scanf("%c", &str[j]);
                    if(str[j] == '\n'){
                        str[j] = '\0';
                        break;
                    }
                }
                i = 0;
                if((str[i] == '-') && (str[i+1] == '1')){
                    printf("End\n\n");
                    for(j = 0; j < L; j++)
                        str[j] = '\0';
                    break;
                }
                else if((str[i] == 'p') && (str[i+1] == 'o') && (str[i+2] == 'p')){
                    if(top == NULL)
                        printf("No Ball\n");
                    else{
                        i += 3;
                        tmp = top;
                        c = 0;
                        while(tmp != NULL){
                            c++;
                            tmp = tmp->next;
                        }
                        if(str[i] == '\0'){
                            printf("pop the last ball %c\n", top->data);
                            pop(&top);
```

```c
                            pop(&top);
                            printf("Box: ");
                            if(top == NULL)
                                printf("NULL\n");
                            else{
                                op(top->next);
                                printf("%c\n", top->data);
                            }
                        }
                        else if(str[i] == 'o'){
                            i += 4;
                            n = 0;
                            while(str[i] != '\0'){
                                n *= 10;
                                n += str[i] - '0';
                                i++;
                            }
                            if(n > c || n < 1){
                                printf("wrong number\n");
                                printf("Box: ");
                                op(top->next);
                                printf("%c\n", top->data);
                            }
                            else{
                                tmp = top;
                                cc = c - n;
                                while(cc--)
                                    tmp = tmp->next;
                                printf("pop number %d ball %c\n", n, tmp->data);
                                printf("Box: ");
                                de(&top, tmp->data);
                                if(top == NULL)
                                    printf("NULL\n");
                                else{
                                    op(top->next);
                                    printf("%c\n", top->data);
                                }
                            }
                        }
                    }
                }
                else if((str[i] == 'p') && (str[i+1] == 'u') && (str[i+2] == 's') && (str[i+3] == 'h')){
                    i += 4;
                    tmp = top;
                    c = 0;
```

```
                    while(tmp != NULL){
                        c++;
                        tmp = tmp->next;
                    }
                    if(str[i] == ' '){
                        printf("push %c ball to the last location\n", str[i+1]);
                        push(&top, str[i+1]);
                        printf("Box: ");
                        op(top->next);
                        printf("%c\n", top->data);
                    }
                    else if(str[i] == 'i'){
                        i += 3;
                        n = 0;
                        while(str[i] != ' '){
                            n *= 10;
                            n += str[i] - '0';
                            i++;
                        }
                        if(n > c+1 || n < 1){
                            i++;
                            printf("push %c ball to the last location\n", str[i]);
                            push(&top, str[i]);
                            printf("Box: ");
                            op(top->next);
                            printf("%c\n", top->data);
                        }
                        else if(n == c+1){
                            i++;
                            printf("push %c ball to number %d location\n", str[i], n);
                            printf("Box: ");
                            push(&top, str[i]);
                            op(top->next);
                            printf("%c\n", top->data);
                        }
                        else{
                            i++;
                            printf("push %c ball to number %d location\n", str[i], n);
                            printf("Box: ");
                            in(&top, str[i], n);
                            op(top->next);
                            printf("%c\n", top->data);
                        }
                    }
                }
```

```
                printf("\nCommand: ");
                for(j = 0; j < L; j++)
                    str[j] = '\0';
            }

        }
        else if(str[i] == 'Q'){
            for(j = 0; j < L; j++)
                str[j] = '\0';
            printf("\nCommand: ");
            while(1){
                for(j = 0; j < L; j++){
                    scanf("%c", &str[j]);
                    if(str[j] == '\n'){
                        str[j] = '\0';
                        break;
                    }
                }
                i = 0;
                if((str[i] == '-') && (str[i+1] == '1')){
                    printf("End\n\n");
                    for(j = 0; j < L; j++)
                        str[j] = '\0';
                    break;
                }
                else if((str[i] == 'p') && (str[i+1] == 'o') && (str[i+2] == 'p')){
                    if(head == NULL)
                        printf("No Ball\n");
                    else{
                        i += 3;
                        tmp2 = head;
                        c = 0;
                        while(tmp2 != NULL){
                            c++;
                            tmp2 = tmp2->next;
                        }
                        if(str[i] == '\0'){
                            printf("pop the first ball %c\n", head->data);
                            dequence(&head, &tail);
                            printf("Box: ");
                            if(head == NULL)
                                printf("NULL\n");
                            else{
                                op2(tail->prev);
                                printf("%c\n", tail->data);
                            }
```

```
                    }
                    else if(str[i] == 'o'){
                        i += 4;
                        n = 0;
                        while(str[i] != '\0'){
                            n *= 10;
                            n += str[i] - '0';
                            i++;
                        }
                        if(n > c || n < 1){
                            printf("wrong number\n");
                            printf("Box: ");
                            op2(tail->prev);
                            printf("%c\n", tail->data);
                        }
                        else{
                            if(n == 1){
                                printf("pop number 1 ball %c\n", head->data);
                                dequence(&head, &tail);
                            }
                            else{
                                if(c == n){
                                    printf("pop number %d ball %c\n", n, tail->data);
                                    de2(&head, &tail, tail->data);
                                }
                                else{
                                    tmp2 = head;
                                    cc = c - n;
                                    while(cc--)
                                        tmp2 = tmp2->next;
                                    printf("pop number %d ball %c\n", n, tmp2->data);
                                    de2(&head, &tail, tmp2->data);
                                }
                            }
                            printf("Box: ");
                            if(head == NULL)
                                printf("NULL\n");
                            else{
                                op2(tail->prev);
                                printf("%c\n", tail->data);
                            }
                        }
                    }
                }
            }
```

```
        else if((str[i] == 'p') && (str[i+1] == 'u') && (str[i+2] == 's') && (str[i+3] == 'h')){
            i += 4;
            tmp2 = head;
            c = 0;
            while(tmp2 != NULL){
                c++;
                tmp2 = tmp2->next;
            }
            if(str[i] == ' '){
                printf("push %c ball to the last location\n", str[i+1]);
                printf("Box: ");
                enquence(&head, &tail, str[i+1]);
                op2(tail->prev);
                printf("%c\n", tail->data);
            }
            else if(str[i] == 'i'){
                i += 3;
                n = 0;
                while(str[i] != ' '){
                    n *= 10;
                    n += str[i] - '0';
                    i++;
                }
                if(n > c+1 || n < 1){
                    i++;
                    printf("push %c ball to the last location\n", str[i]);
                    printf("Box: ");
                    enquence(&head, &tail, str[i]);
                    op2(tail->prev);
                    printf("%c\n", tail->data);
                }
                else if(n == c+1){
                    i++;
                    printf("push %c ball to number %d location\n", str[i], n);
                    printf("Box: ");
                    enquence(&head, &tail, str[i]);
                    op2(tail->prev);
                    printf("%c\n", tail->data);
                }
                else{
                    i++;
                    printf("push %c ball to number %d location\n", str[i], n);
                    printf("Box: ");
                    in2(&head, &tail, str[i], n);
                    op2(tail->prev);
```

```
                            op2(tail->prev);
                            printf("%c\n", tail->data);
                        }
                    }
                }
                printf("\nCommand: ");
                for(j = 0; j < L; j++)
                    str[j] = '\0';
            }
        }
        printf("please enter sequence: ");
    }

    return 0;
}

void push(NODE1** TOP, char value){
    NODE1 *newp;

    newp = (NODE1*)malloc(sizeof(NODE1));
    newp->data = value;
    newp->next = NULL;
    newp->next = *TOP;
    *TOP = newp;
}

void pop(NODE1** TOP){
    NODE1 *temp;
    char value;

    temp = *TOP;
    *TOP = (*TOP)->next;
    value = temp->data;
    free(temp);
}

void in(NODE1** TOP, char value, int n){
    NODE1 *newp, *prev, *current;

    newp = (NODE1*)malloc(sizeof(NODE1));
    newp->data = value;
    newp->next = NULL;
    prev = NULL;
    current = *TOP;
    while(current != NULL && n > 0){
```

```
358     while(current != NULL && n > 0){
359         prev = current;
360         current = current->next;
361         n--;
362     }
363     if(prev == NULL){
364         newp->next = *TOP;
365         *TOP = newp;
366     }
367     else{
368         prev->next = newp;
369         newp->next = current;
370     }
371 }
372
373 void de(NODE1** TOP, char value){
374     NODE1 *prev, *current, *tmp;
375
376     if(value == (*TOP)->data){
377         tmp = *TOP;
378         *TOP = (*TOP)->next;
379         free(tmp);
380     }
381     else{
382         prev = *TOP;
383         current = (*TOP)->next;
384         while(current != NULL && current->data != value){
385         prev = current;
386         current = current->next;
387         }
388         if(current != NULL){
389             tmp = current;
390             prev->next = current->next;
391             free(tmp);
392         }
393     }
394 }
395
396 int op(NODE1* TOPn){
397     if(TOPn == NULL)
398         return 0;
399     else if(TOPn->next == NULL)
400         printf("%c->",TOPn->data);
401     else{
402         op(TOPn->next);
```

```
403         printf("%c->",TOPn->data);
404     }
405 }
406
407 void enquence(NODE2** head, NODE2** tail, char value){
408     NODE2 *newp;
409
410     newp = (NODE2*)malloc(sizeof(NODE2));
411     newp->data = value;
412     newp->next = NULL;
413     newp->prev = NULL;
414     if(*head == NULL)
415         *head = newp;
416     else
417         (*tail)->next = newp;
418     newp->prev = *tail;
419     *tail = newp;
420 }
421
422 void dequence(NODE2** head, NODE2** tail){
423     NODE2 *temp;
424
425     temp = *head;
426     *head = (*head)->next;
427     if(*head == NULL)
428         *tail = NULL;
429     else
430         (*head)->prev = NULL;
431     free(temp);
432 }
433
434 void in2(NODE2** head, NODE2** tail, char value, int n){
435     NODE2 *newp, *p, *c;
436
437     newp = (NODE2*)malloc(sizeof(NODE2));
438     newp->data = value;
439     newp->next = NULL;
440     newp->prev = NULL;
441     p = NULL;
442     c = *head;
443     while(c != NULL && n > 0){
444         p = c;
445         c = c->next;
446         n--;
447     }
```

```
448     if(p == NULL){
449         newp->prev = *tail;
450         *tail = newp;
451     }
452     else{
453         p->prev->next = newp;
454         newp->prev = p->prev;
455         p->prev = newp;
456         newp->next = p;
457     }
458 }
459
460 void de2(NODE2** head, NODE2** tail, char value){
461     NODE2 *p, *c, *tmp;
462
463     if(value == (*head)->data){
464         tmp = *head;
465         *head = (*head)->next;
466         if(*head != NULL)
467             (*head)->prev = NULL;
468         free(tmp);
469     }
470     else if(value == (*tail)->data){
471         tmp = *tail;
472         *tail = (*tail)->prev;
473         (*tail)->next = NULL;
474         free(tmp);
475     }
476     else{
477         p = *head;
478         c = (*head)->next;
479         while(c != NULL && c->data != value){
480             p = c;
481             c = c->next;
482         }
483         if(c != NULL){
484             tmp = c;
485             p->next = c->next;
486             c->next->prev = p;
487             free(tmp);
488         }
489     }
490 }
491
```

```
492 int op2(NODE2* tailn){
493     if(tailn == NULL)
494         return 0;
495     else if(tailn->prev == NULL)
496         printf("%c->",tailn->data);
497     else{
498         op2(tailn->prev);
499         printf("%c->",tailn->data);
500     }
501 }
```

Kiuwan

	Files	Defects	Rule	Priority	Characteristic	Language	Effort
Σ		157					73h 44
▸	1	7	CERT C MEM00: Allocate and free memory in the same module at the same level of abstraction		Security	C	3h 30
▸	1	4	Allocated memory must be released in same scope		Efficiency	C	2h 00
▸	1	4	CERT C MEM32: Detect and handle memory allocation errors		Security	C	2h 00
▸	1	2	MISRA 16.2: Functions shall not call themselves, either directly or indirectly		Reliability	C	8h 00
▸	1	1	MISRA 9.2: Braces shall be used to indicate and match the structure of the non-zero initialisation of arrays and structures		Reliability	C	30m
▸	1	62	CERT C FIO33: Detect and handle input/output errors resulting in undefined behavior		Security	C	31h 00
▸	1	14	MISRA 12.2: The value of an expression shall be the same under any order of evaluation that the standard permits		Reliability	C	7h 00
▸	1	10	MISRA 16.3: Names shall be given for all parameters in function prototype		Maintainability	C	30m
▸	1	8	MISRA 12.5: The operands of a logical && or || shall be primary-expressions		Reliability	C	48m
▸	1	1	Avoid functions and methods with too many lines of code		Maintainability	C	4h 00
▸	1	1	MISRA 16.5: Functions with no parameters shall be declared with parameter type void		Maintainability	C	03m
▸	1	14	MISRA 14.9: If-else statements must use braces		Reliability	C	1h 24
▸	1	10	Formal parameters names in function definition and declaration		Maintainability	C	30m
▸	1	8	MISRA 14.8: Loops must use braces to delimit loop body		Reliability	C	48m
▸	1	7	MISRA 14.10: All if...else if constructs shall be terminated with an else clause		Maintainability	C	3h 30
▸	1	2	Avoid classes, structs or unions with low comment/code ratio		Maintainability	C	12m
▸	1	2	MISRA 13.2: Tests of a value against zero should be made explicit, unless the operand is effectively Boolean		Maintainability	C	8h 00

Splint

```
wei@wei-virtual-machine:~/Desktop$ splint B043040003.c +bounds -paramuse -varuse
Splint 3.1.2 --- 03 May 2009

B043040003.c: (in function main)
B043040003.c:31:17: Initializer block for str has 1 element, but declared as
                      char [20]: '\0'
  Initializer does not define all elements of a declared array. (Use
  -initallelements to inhibit warning)
B043040003.c:38:3: Operand of ++ is non-numeric (char): voc
  Types are incompatible. (Use -type to inhibit warning)
B043040003.c:43:3: Operand of ++ is non-numeric (char): voc
B043040003.c:46:8: Test expression for while not boolean, type int: 1
  Test expression type is not boolean or int. (Use -predboolint to inhibit
  warning)
B043040003.c:48:4: Return value (type int) ignored: scanf("%c", &str[j])
  Result returned by function call is not used. If this is intended, can cast
  result to (void) to eliminate message. (Use -retvalint to inhibit warning)
B043040003.c:61:10: Test expression for while not boolean, type int: 1
B043040003.c:63:6: Return value (type int) ignored: scanf("%c", &str[j])
B043040003.c:94:9: Return value (type int) ignored: op(top->next)
B043040003.c:103:9: Incompatible types for += (int, char): n += str[i] - '0'
  To make char and int types equivalent, use +charint.
B043040003.c:109:9: Return value (type int) ignored: op(top->next)
B043040003.c:115:17: Test expression for while not boolean, type int: cc--
B043040003.c:123:10: Return value (type int) ignored: op(top->next)
B043040003.c:142:13: Arrow access from null pointer top: top->next
  A possibly null pointer is dereferenced.  Value is either the result of a
  function which may return null (in which case, code should check it is not
  null), or a global, parameter or structure field declared with the null
  qualifier. (Use -nullderef to inhibit warning)
   B043040003.c:33:15: Storage top becomes null
B043040003.c:142:7: Return value (type int) ignored: op(top->next)
B043040003.c:150:8: Incompatible types for += (int, char): n += str[i] - '0'
B043040003.c:158:14: Arrow access from null pointer top: top->next
   B043040003.c:33:15: Storage top becomes null
B043040003.c:158:8: Return value (type int) ignored: op(top->next)
B043040003.c:166:14: Arrow access from null pointer top: top->next
   B043040003.c:33:15: Storage top becomes null
B043040003.c:166:8: Return value (type int) ignored: op(top->next)
B043040003.c:174:14: Arrow access from null pointer top: top->next
   B043040003.c:33:15: Storage top becomes null
B043040003.c:174:8: Return value (type int) ignored: op(top->next)
B043040003.c:189:10: Test expression for while not boolean, type int: 1
B043040003.c:191:6: Return value (type int) ignored: scanf("%c", &str[j])
B043040003.c:222:17: Arrow access from null pointer tail: tail->prev
   B043040003.c:34:16: Storage tail becomes null
B043040003.c:222:9: Return value (type int) ignored: op2(tail->prev)
B043040003.c:231:9: Incompatible types for += (int, char): n += str[i] - '0'
B043040003.c:237:17: Arrow access from null pointer tail: tail->prev
```

```
B043040003.c:237:17: Arrow access from null pointer tail: tail->prev
   B043040003.c:34:16: Storage tail becomes null
B043040003.c:237:9: Return value (type int) ignored: op2(tail->prev)
B043040003.c:247:52: Arrow access from null pointer tail: tail->data
   B043040003.c:34:16: Storage tail becomes null
B043040003.c:253:19: Test expression for while not boolean, type int: cc--
B043040003.c:263:18: Arrow access from null pointer tail: tail->prev
   B043040003.c:34:16: Storage tail becomes null
B043040003.c:263:10: Return value (type int) ignored: op2(tail->prev)
B043040003.c:282:15: Arrow access from null pointer tail: tail->prev
   B043040003.c:34:16: Storage tail becomes null
B043040003.c:282:7: Return value (type int) ignored: op2(tail->prev)
B043040003.c:290:8: Incompatible types for += (int, char): n += str[i] - '0'
B043040003.c:298:16: Arrow access from null pointer tail: tail->prev
   B043040003.c:34:16: Storage tail becomes null
B043040003.c:298:8: Return value (type int) ignored: op2(tail->prev)
B043040003.c:306:16: Arrow access from null pointer tail: tail->prev
   B043040003.c:34:16: Storage tail becomes null
B043040003.c:306:8: Return value (type int) ignored: op2(tail->prev)
B043040003.c:314:16: Arrow access from null pointer tail: tail->prev
   B043040003.c:34:16: Storage tail becomes null
B043040003.c:314:8: Return value (type int) ignored: op2(tail->prev)
B043040003.c:55:26: Likely out-of-bounds read: str[i + 1]
    Unable to resolve constraint:
    requires 0 >= 1
     needed to satisfy precondition:
    requires maxRead(str @ B043040003.c:55:26) >= i @ B043040003.c:55:30 + 1
  A memory read references memory beyond the allocated storage. (Use
  -likelyboundsread to inhibit warning)
B043040003.c:191:19: Possible out-of-bounds read: str[j]
    Unable to resolve constraint:
    requires j @ B043040003.c:191:23 <= 0
     needed to satisfy precondition:
    requires maxRead(str @ B043040003.c:191:19) >= j @ B043040003.c:191:23
  A memory read references memory beyond the allocated storage. (Use
  -boundsread to inhibit warning)
B043040003.c:311:55: Likely out-of-bounds read: str[i]
    Unable to resolve constraint:
    requires 0 >= 8
     needed to satisfy precondition:
    requires maxRead(str @ B043040003.c:311:55) >= i @ B043040003.c:311:59
B043040003.c: (in function push)
B043040003.c:334:6: Arrow access from possibly null pointer newp: newp->data
   B043040003.c:333:9: Storage newp may become null
B043040003.c:336:2: Unqualified storage *TOP assigned to implicitly only:
                      newp->next = *TOP
  Unqualified storage is transferred in an inconsistent way. (Use
  -unqualifiedtrans to inhibit warning)
B043040003.c:338:2: Fresh storage newp not released before return
```

```
B043040003.c:338:2: Fresh storage newp not released before return
  A memory leak has been detected. Storage allocated locally is not released
  before the last reference to it is lost. (Use -mustfreefresh to inhibit
  warning)
   B043040003.c:333:2: Fresh storage newp created
B043040003.c:338:2: Storage *TOP reachable from parameter is kept (should be
                      unqualified)
  Storage derivable from a parameter does not match the alias kind expected for
  the formal parameter. (Use -compmempass to inhibit warning)
   B043040003.c:336:2: Storage *TOP becomes kept
B043040003.c:336:15: Possible out-of-bounds read: *TOP
    Unable to resolve constraint:
    requires maxRead(TOP @ B043040003.c:336:16) >= 0
     needed to satisfy precondition:
    requires maxRead(TOP @ B043040003.c:336:16) >= 0
B043040003.c:337:2: Possible out-of-bounds store: *TOP
    Unable to resolve constraint:
    requires maxSet(TOP @ B043040003.c:337:3) >= 0
     needed to satisfy precondition:
    requires maxSet(TOP @ B043040003.c:337:3) >= 0
  A memory write may write to an address beyond the allocated buffer. (Use
  -boundswrite to inhibit warning)
B043040003.c: (in function pop)
B043040003.c:345:2: Only storage *TOP->next assigned to unqualified:
                      *TOP = (*TOP)->next
  The only reference to this storage is transferred to another reference (e.g.,
  by returning it) that does not have the only annotation. This may lead to a
  memory leak, since the new reference is not necessarily released. (Use
  -onlytrans to inhibit warning)
B043040003.c:347:7: Only storage temp->next (type struct node1 *) derived from
                      released storage is not released (memory leak): temp
  A storage leak due to incomplete deallocation of a structure or deep pointer
  is suspected. Unshared storage that is reachable from a reference that is
  being deallocated has not yet been deallocated. Splint assumes when an object
  is passed as an out only void pointer that the outer object will be
  deallocated, but the inner objects will not. (Use -compdestroy to inhibit
  warning)
B043040003.c:345:2: Possible out-of-bounds store: *TOP
    Unable to resolve constraint:
    requires maxSet(TOP @ B043040003.c:345:3) >= 0
     needed to satisfy precondition:
    requires maxSet(TOP @ B043040003.c:345:3) >= 0
B043040003.c:345:10: Possible out-of-bounds read: *TOP
    Unable to resolve constraint:
    requires maxRead(TOP @ B043040003.c:345:11) >= 0
     needed to satisfy precondition:
    requires maxRead(TOP @ B043040003.c:345:11) >= 0
B043040003.c: (in function in)
B043040003.c:354:6: Arrow access from possibly null pointer newp: newp->data
```

```
B043040003.c:354:6: Arrow access from possibly null pointer newp: newp->data
   B043040003.c:353:9: Storage newp may become null
B043040003.c:364:3: Unqualified storage *TOP assigned to implicitly only:
                      newp->next = *TOP
B043040003.c:368:3: Implicitly only storage prev->next (type struct node1 *)
                      not released before assignment: prev->next = newp
  A memory leak has been detected. Only-qualified storage is not released
  before the last reference to it is lost. (Use -mustfreeonly to inhibit
  warning)
B043040003.c:370:2: Variable newp is kept in false branch, but not kept in true
                      branch.
  The state of a variable is different depending on which branch is taken. This
  means no annotation can sensibly be applied to the storage. (Use -branchstate
  to inhibit warning)
   B043040003.c:370:2: in false branch:
   B043040003.c:368:3: Storage newp becomes kept
   B043040003.c:370:2: in true branch:
   B043040003.c:353:2: Fresh storage newp created
B043040003.c:371:2: Storage *TOP reachable from parameter is kept (should be
                      unqualified)
   B043040003.c:369:3: Storage *TOP becomes kept
B043040003.c:364:16: Possible out-of-bounds read: *TOP
    Unable to resolve constraint:
    requires maxRead(TOP @ B043040003.c:364:17) >= 0
     needed to satisfy precondition:
    requires maxRead(TOP @ B043040003.c:364:17) >= 0
B043040003.c:365:3: Possible out-of-bounds store: *TOP
    Unable to resolve constraint:
    requires maxSet(TOP @ B043040003.c:365:4) >= 0
     needed to satisfy precondition:
    requires maxSet(TOP @ B043040003.c:365:4) >= 0
B043040003.c: (in function de)
B043040003.c:378:3: Only storage *TOP->next assigned to unqualified:
                      *TOP = (*TOP)->next
B043040003.c:379:8: Only storage tmp->next (type struct node1 *) derived from
                      released storage is not released (memory leak): tmp
B043040003.c:390:4: Implicitly only storage prev->next (type struct node1 *)
    not released before assignment: prev->next = current->next
B043040003.c:391:9: Kept storage tmp passed as only param: free (tmp)
  storage is transferred to a non-temporary reference after being passed as
  keep parameter. The storage may be released or new aliases created. (Use
  -kepttrans to inhibit warning)
   B043040003.c:390:4: Storage tmp becomes kept
B043040003.c:392:3: Storage prev->next->next is kept in one path, but live in
                      another.
   B043040003.c:390:4: Storage prev->next->next becomes kept
B043040003.c:393:2: Storage *TOP->next is released in one path, but live in
                      another.
   B043040003.c:391:9: Storage *TOP->next released
```

```
   B043040003.c:391:9: Storage *TOP->next released
B043040003.c:376:15: Possible out-of-bounds read: *TOP
    Unable to resolve constraint:
    requires maxRead(TOP @ B043040003.c:376:16) >= 0
     needed to satisfy precondition:
    requires maxRead(TOP @ B043040003.c:376:16) >= 0
B043040003.c:378:3: Possible out-of-bounds store: *TOP
    Unable to resolve constraint:
    requires maxSet(TOP @ B043040003.c:378:4) >= 0
     needed to satisfy precondition:
    requires maxSet(TOP @ B043040003.c:378:4) >= 0
B043040003.c: (in function op)
B043040003.c:402:3: Return value (type int) ignored: op(TOPn->next)
B043040003.c:405:2: Path with no return in function declared to return int
  There is a path through a function declared to return a value on which there
  is no return statement. This means the execution may fall through without
  returning a meaningful result to the caller. (Use -noret to inhibit warning)
B043040003.c: (in function enquence)
B043040003.c:411:6: Arrow access from possibly null pointer newp: newp->data
   B043040003.c:410:9: Storage newp may become null
B043040003.c:417:3: Implicitly only storage *tail->next (type struct node2 *)
                      not released before assignment: (*tail)->next = newp
B043040003.c:418:2: Unqualified storage *tail assigned to implicitly only:
                      newp->prev = *tail
B043040003.c:420:2: Storage *tail reachable from parameter is kept (should be
                      unqualified)
   B043040003.c:418:2: Storage *tail becomes kept
B043040003.c:420:2: Function returns with null storage derivable from parameter
                      *tail->next
  A possibly null pointer is reachable from a parameter or global variable that
  is not declared using a /*@null@*/ annotation. (Use -nullstate to inhibit
  warning)
   B043040003.c:412:15: Storage *tail->next becomes null
B043040003.c:420:2: Function returns with null storage derivable from parameter
                      *tail->prev->next->next
   B043040003.c:412:15: Storage *tail->prev->next->next becomes null
B043040003.c:420:2: Function returns with null storage derivable from parameter
                      *tail->prev->next->prev
   B043040003.c:413:15: Storage *tail->prev->next->prev becomes null
B043040003.c:414:5: Possible out-of-bounds read: *head
    Unable to resolve constraint:
    requires maxRead(head @ B043040003.c:414:6) >= 0
     needed to satisfy precondition:
    requires maxRead(head @ B043040003.c:414:6) >= 0
B043040003.c:415:3: Possible out-of-bounds store: *head
    Unable to resolve constraint:
    requires maxSet(head @ B043040003.c:415:4) >= 0
     needed to satisfy precondition:
    requires maxSet(head @ B043040003.c:415:4) >= 0
```

```
    requires maxSet(head @ B043040003.c:415:4) >= 0
B043040003.c:418:15: Possible out-of-bounds read: *tail
    Unable to resolve constraint:
    requires maxRead(tail @ B043040003.c:418:16) >= 0
     needed to satisfy precondition:
    requires maxRead(tail @ B043040003.c:418:16) >= 0
B043040003.c:419:2: Possible out-of-bounds store: *tail
    Unable to resolve constraint:
    requires maxSet(tail @ B043040003.c:419:3) >= 0
     needed to satisfy precondition:
    requires maxSet(tail @ B043040003.c:419:3) >= 0
B043040003.c: (in function dequence)
B043040003.c:426:2: Only storage *head->next assigned to unqualified:
                      *head = (*head)->next
B043040003.c:431:7: Only storage temp->next (type struct node2 *) derived from
                      released storage is not released (memory leak): temp
B043040003.c:431:7: Only storage temp->prev (type struct node2 *) derived from
                      released storage is not released (memory leak): temp
B043040003.c:432:2: Function returns with null storage derivable from parameter
                      *head->next->next->prev
   B043040003.c:430:19: Storage *head->next->next->prev becomes null
B043040003.c:432:2: Function returns with null storage derivable from parameter
                      *head->next->prev
   B043040003.c:430:19: Storage *head->next->prev becomes null
B043040003.c:432:2: Function returns with null storage derivable from parameter
                      *tail
   B043040003.c:428:11: Storage *tail becomes null
B043040003.c:427:5: Possible out-of-bounds read: *head
    Unable to resolve constraint:
    requires maxRead(head @ B043040003.c:427:6) >= 0
     needed to satisfy precondition:
    requires maxRead(head @ B043040003.c:427:6) >= 0
B043040003.c:428:3: Possible out-of-bounds store: *tail
    Unable to resolve constraint:
    requires maxSet(tail @ B043040003.c:428:4) >= 0
     needed to satisfy precondition:
    requires maxSet(tail @ B043040003.c:428:4) >= 0
B043040003.c:430:4: Possible out-of-bounds store: *head
    Unable to resolve constraint:
    requires maxSet(head @ B043040003.c:430:5) >= 0
     needed to satisfy precondition:
    requires maxSet(head @ B043040003.c:430:5) >= 0
B043040003.c: (in function in2)
B043040003.c:438:6: Arrow access from possibly null pointer newp: newp->data
   B043040003.c:437:9: Storage newp may become null
B043040003.c:449:3: Unqualified storage *tail assigned to implicitly only:
                      newp->prev = *tail
B043040003.c:453:3: Implicitly only storage p->prev->next (type struct node2 *)
                      not released before assignment: p->prev->next = newp
```

```
                      not released before assignment: p->prev->next = newp
B043040003.c:455:3: Kept storage newp assigned to implicitly only:
                      p->prev = newp
   B043040003.c:453:3: Storage newp becomes kept
B043040003.c:457:2: Variable newp is kept in false branch, but not kept in true
                      branch.
   B043040003.c:457:2: in false branch:
   B043040003.c:453:3: Storage newp becomes kept
   B043040003.c:457:2: in true branch:
   B043040003.c:437:2: Fresh storage newp created
B043040003.c:457:2: Storage *tail is kept in one path, but live in another.
   B043040003.c:449:3: Storage *tail becomes kept
B043040003.c:458:2: Storage *head reachable from parameter is kept (should be
                      unqualified)
   B043040003.c:456:3: Storage *head becomes kept
B043040003.c:458:2: Function returns with null storage derivable from parameter
                      *head->prev->next
   B043040003.c:439:15: Storage *head->prev->next becomes null
B043040003.c:458:2: Function returns with null storage derivable from parameter
                      *head->prev->prev->next->next
   B043040003.c:439:15: Storage *head->prev->prev->next->next becomes null
B043040003.c:458:2: Function returns with null storage derivable from parameter
                      *head->prev->prev->next->prev
   B043040003.c:440:15: Storage *head->prev->prev->next->prev becomes null
B043040003.c:458:2: Storage *tail reachable from parameter is kept (should be
                      unqualified)
   B043040003.c:449:3: Storage *tail becomes kept
B043040003.c:458:2: Function returns with null storage derivable from parameter
                      *tail->next
   B043040003.c:439:15: Storage *tail->next becomes null
B043040003.c:442:6: Possible out-of-bounds read: *head
    Unable to resolve constraint:
    requires maxRead(head @ B043040003.c:442:7) >= 0
     needed to satisfy precondition:
    requires maxRead(head @ B043040003.c:442:7) >= 0
B043040003.c:449:16: Possible out-of-bounds read: *tail
    Unable to resolve constraint:
    requires maxRead(tail @ B043040003.c:449:17) >= 0
     needed to satisfy precondition:
    requires maxRead(tail @ B043040003.c:449:17) >= 0
B043040003.c:450:3: Possible out-of-bounds store: *tail
    Unable to resolve constraint:
    requires maxSet(tail @ B043040003.c:450:4) >= 0
     needed to satisfy precondition:
    requires maxSet(tail @ B043040003.c:450:4) >= 0
B043040003.c: (in function de2)
B043040003.c:465:3: Only storage *head->next assigned to unqualified:
                      *head = (*head)->next
B043040003.c:468:8: Only storage tmp->next (type struct node2 *) derived from
```

```
B043040003.c:468:8: Only storage tmp->next (type struct node2 *) derived from
                      released storage is not released (memory leak): tmp
B043040003.c:468:8: Only storage tmp->prev (type struct node2 *) derived from
                      released storage is not released (memory leak): tmp
B043040003.c:472:3: Only storage *tail->prev assigned to unqualified:
                      *tail = (*tail)->prev
B043040003.c:474:8: Only storage tmp->next (type struct node2 *) derived from
                      released storage is not released (memory leak): tmp
B043040003.c:474:8: Only storage tmp->prev (type struct node2 *) derived from
                      released storage is not released (memory leak): tmp
B043040003.c:485:4: Implicitly only storage p->next (type struct node2 *) not
                      released before assignment: p->next = c->next
B043040003.c:487:9: Kept storage *head->next passed as only param (tmp aliases
                      *head->next): free (tmp)
   B043040003.c:486:4: Storage *head->next becomes kept
B043040003.c:487:9: Kept storage tmp passed as only param: free (tmp)
   B043040003.c:485:4: Storage tmp becomes kept
B043040003.c:488:3: Storage *head is kept in one path, but live in another.
   B043040003.c:486:4: Storage *head becomes kept
B043040003.c:489:2: Storage *head->next is released in one path, but live in
                      another.
   B043040003.c:487:9: Storage *head->next released
B043040003.c:490:2: Function returns with null storage derivable from parameter
                      *tail->prev->prev->next
   B043040003.c:473:19: Storage *tail->prev->prev->next becomes null
B043040003.c:490:2: Function returns with null storage derivable from parameter
                      *tail->prev->next
   B043040003.c:473:19: Storage *tail->prev->next becomes null
B043040003.c:463:15: Possible out-of-bounds read: *head
    Unable to resolve constraint:
    requires maxRead(head @ B043040003.c:463:16) >= 0
     needed to satisfy precondition:
    requires maxRead(head @ B043040003.c:463:16) >= 0
B043040003.c:467:5: Possible out-of-bounds store: *head
    Unable to resolve constraint:
    requires maxSet(head @ B043040003.c:467:6) >= 0
     needed to satisfy precondition:
    requires maxSet(head @ B043040003.c:467:6) >= 0
B043040003.c:470:20: Possible out-of-bounds read: *tail
    Unable to resolve constraint:
    requires maxRead(tail @ B043040003.c:470:21) >= 0
     needed to satisfy precondition:
    requires maxRead(tail @ B043040003.c:470:21) >= 0
B043040003.c:473:4: Possible out-of-bounds store: *tail
    Unable to resolve constraint:
    requires maxSet(tail @ B043040003.c:473:5) >= 0
     needed to satisfy precondition:
    requires maxSet(tail @ B043040003.c:473:5) >= 0
B043040003.c: (in function op2)
```

```
B043040003.c: (in function op2)
B043040003.c:498:3: Return value (type int) ignored: op2(tailn->prev)
B043040003.c:501:2: Path with no return in function declared to return int
B043040003.c:17:6: Function exported but not used outside B043040003: push
  A declaration is exported, but not used outside this module. Declaration can
  use static qualifier. (Use -exportlocal to inhibit warning)
   B043040003.c:338:1: Definition of push
B043040003.c:18:6: Function exported but not used outside B043040003: pop
   B043040003.c:348:1: Definition of pop
B043040003.c:19:6: Function exported but not used outside B043040003: in
   B043040003.c:371:1: Definition of in
B043040003.c:20:6: Function exported but not used outside B043040003: de
   B043040003.c:394:1: Definition of de
B043040003.c:21:5: Function exported but not used outside B043040003: op
   B043040003.c:405:1: Definition of op
B043040003.c:23:6: Function exported but not used outside B043040003: enquence
   B043040003.c:420:1: Definition of enquence
B043040003.c:24:6: Function exported but not used outside B043040003: dequence
   B043040003.c:432:1: Definition of dequence
B043040003.c:25:6: Function exported but not used outside B043040003: in2
   B043040003.c:458:1: Definition of in2
B043040003.c:26:6: Function exported but not used outside B043040003: de2
   B043040003.c:490:1: Definition of de2
B043040003.c:27:5: Function exported but not used outside B043040003: op2
   B043040003.c:501:1: Definition of op2

Finished checking --- 135 code warnings
wei@wei-virtual-machine:~/Desktop$
```

陣列用迴圈一個一個初始化比較好
if-else、for、while 最好用括號
沒有用到 op()的回傳值，可以宣告為 void
沒有用到 op2()的回傳值，可以宣告為 void
malloc 完要檢查是否為 NULL，因為有可能會失敗