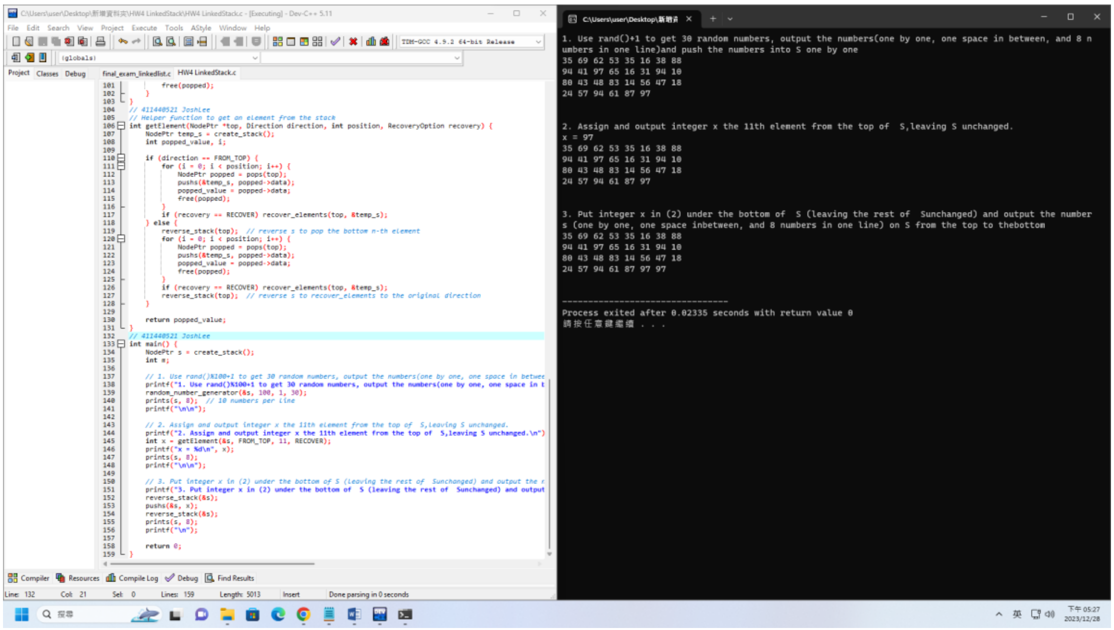
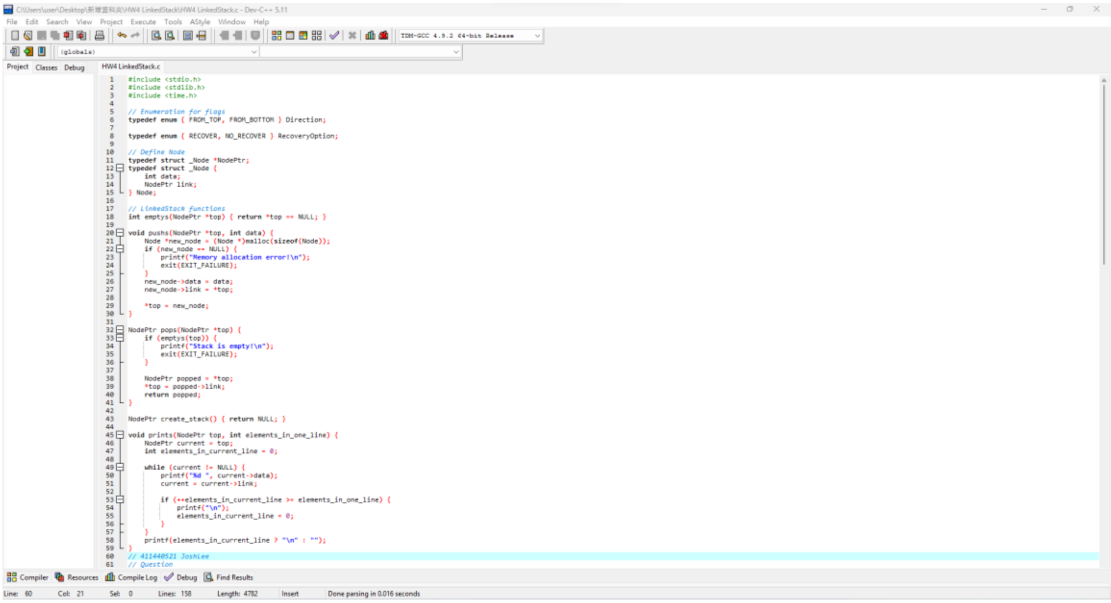
内容

1. Linked Stack

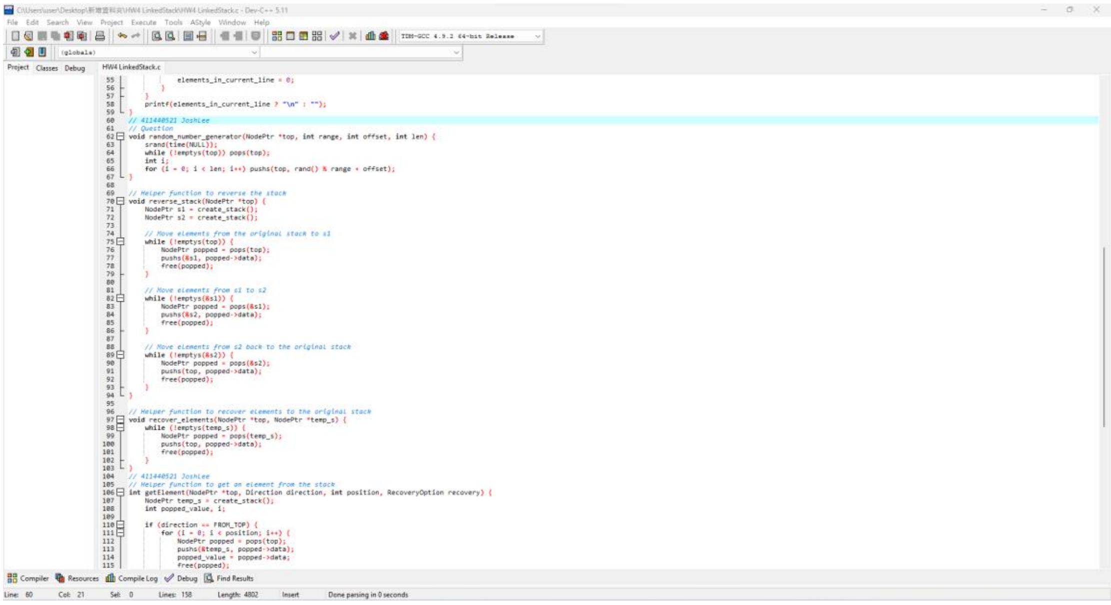

C:\Users\user\Desktop\新增資料夾\HW4 LinkedStack\HW4 LinkedStack.c - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_linkedlist.c
HW4 LinkedStack.c
Compiler Resources Compile Log Debug Find Results
Line: 132 Col: 21 Sel: 0 Lines: 159 Length: 5013 Insert Done parsing in 0 seconds
C:\Users\user\Desktop\新增資
下午 05:27
2023/12/28

```c
            free(popped);
        }
}
// 411440521 JoshLee
// Helper function to get an element from the stack
int getElement(NodePtr *top, Direction direction, int position, RecoveryOption recovery) {
    NodePtr temp_s = create_stack();
    int popped_value, i;

    if (direction == FROM_TOP) {
        for (i = 0; i < position; i++) {
            NodePtr popped = pops(top);
            pushs(&temp_s, popped->data);
            popped_value = popped->data;
            free(popped);
        }
        if (recovery == RECOVER) recover_elements(top, &temp_s);
    } else {
        reverse_stack(top);  // reverse s to pop the bottom n-th element
        for (i = 0; i < position; i++) {
            NodePtr popped = pops(top);
            pushs(&temp_s, popped->data);
            popped_value = popped->data;
            free(popped);
        }
        if (recovery == RECOVER) recover_elements(top, &temp_s);
        reverse_stack(top);  // reverse s to recover_elements to the original direction
    }

    return popped_value;
}
// 411440521 JoshLee
int main() {
    NodePtr s = create_stack();
    int m;

    // 1. Use rand()%100+1 to get 30 random numbers, output the numbers(one by one, one space in betwee
    printf("1. Use rand()%100+1 to get 30 random numbers, output the numbers(one by one, one space in t
    random_number_generator(&s, 100, 1, 30);
    prints(s, 8);  // 10 numbers per line
    printf("\n\n");

    // 2. Assign and output integer x the 11th element from the top of  S,leaving S unchanged.
    printf("2. Assign and output integer x the 11th element from the top of  S,leaving S unchanged.\n")
    int x = getElement(&s, FROM_TOP, 11, RECOVER);
    printf("x = %d\n", x);
    prints(s, 8);
    printf("\n\n");

    // 3. Put integer x in (2) under the bottom of S (leaving the rest of  Sunchanged) and output the r
    printf("3. Put integer x in (2) under the bottom of  S (leaving the rest of  Sunchanged) and output
    reverse_stack(&s);
    pushs(&s, x);
    reverse_stack(&s);
    prints(s, 8);
    printf("\n");

    return 0;
}
```

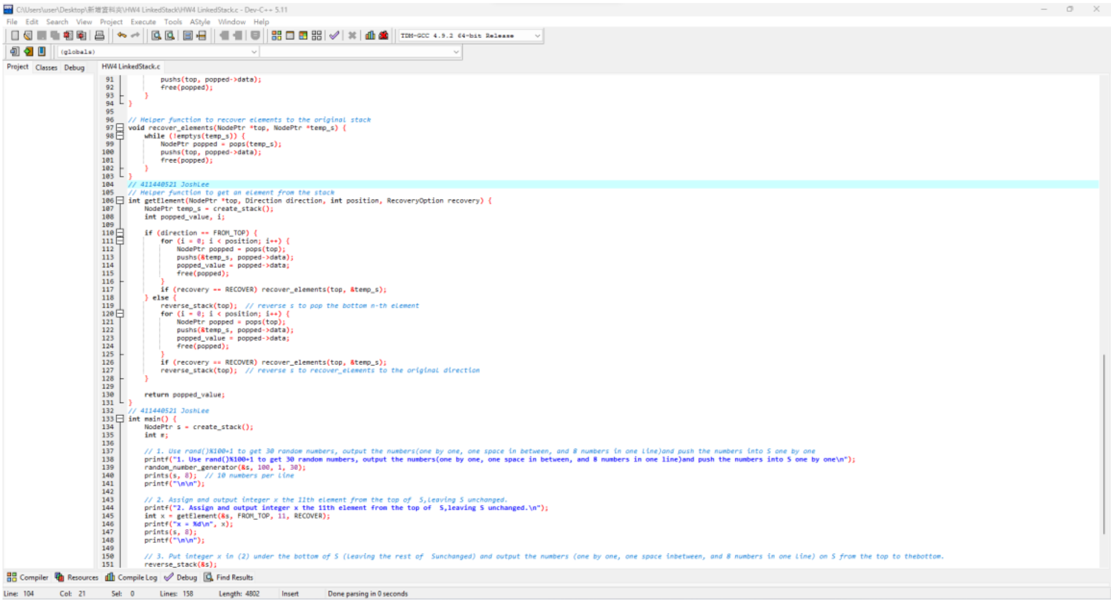
```
1. Use rand()+1 to get 30 random numbers, output the numbers(one by one, one space in between, and 8 n
umbers in one line)and push the numbers into S one by one
35 69 62 53 35 16 38 88
94 41 97 65 16 31 94 10
80 43 48 83 14 56 47 18
24 57 94 61 87 97


2. Assign and output integer x the 11th element from the top of  S,leaving S unchanged.
x = 97
35 69 62 53 35 16 38 88
94 41 97 65 16 31 94 10
80 43 48 83 14 56 47 18
24 57 94 61 87 97


3. Put integer x in (2) under the bottom of  S (leaving the rest of  Sunchanged) and output the number
s (one by one, one space inbetween, and 8 numbers in one line) on S from the top to thebottom
35 69 62 53 35 16 38 88
94 41 97 65 16 31 94 10
80 43 48 83 14 56 47 18
24 57 94 61 87 97 97


--------------------------------
Process exited after 0.02335 seconds with return value 0
請按任意鍵繼續 . . .
```


C:\Users\user\Desktop\新增資料夾\HW4 LinkedStack\HW4 LinkedStack.c - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
HW4 LinkedStack.c
Compiler Resources Compile Log Debug Find Results
Line: 60 Col: 21 Sel: 0 Lines: 158 Length: 4782 Insert Done parsing in 0.016 seconds

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

// Enumeration for flags
typedef enum { FROM_TOP, FROM_BOTTOM } Direction;

typedef enum { RECOVER, NO_RECOVER } RecoveryOption;

// Define Node
typedef struct _Node *NodePtr;
typedef struct _Node {
    int data;
    NodePtr link;
} Node;

// LinkedStack functions
int emptys(NodePtr *top) { return *top == NULL; }

void pushs(NodePtr *top, int data) {
    Node *new_node = (Node *)malloc(sizeof(Node));
    if (new_node == NULL) {
        printf("Memory allocation error!\n");
        exit(EXIT_FAILURE);
    }
    new_node->data = data;
    new_node->link = *top;

    *top = new_node;
}

NodePtr pops(NodePtr *top) {
    if (emptys(top)) {
        printf("Stack is empty!\n");
        exit(EXIT_FAILURE);
    }

    NodePtr popped = *top;
    *top = popped->link;
    return popped;
}

NodePtr create_stack() { return NULL; }

void prints(NodePtr top, int elements_in_one_line) {
    NodePtr current = top;
    int elements_in_current_line = 0;

    while (current != NULL) {
        printf("%d ", current->data);
        current = current->link;

        if (++elements_in_current_line >= elements_in_one_line) {
            printf("\n");
            elements_in_current_line = 0;
        }
    }
    printf(elements_in_current_line ? "\n" : "");
}
// 411440521 JoshLee
// Question
```

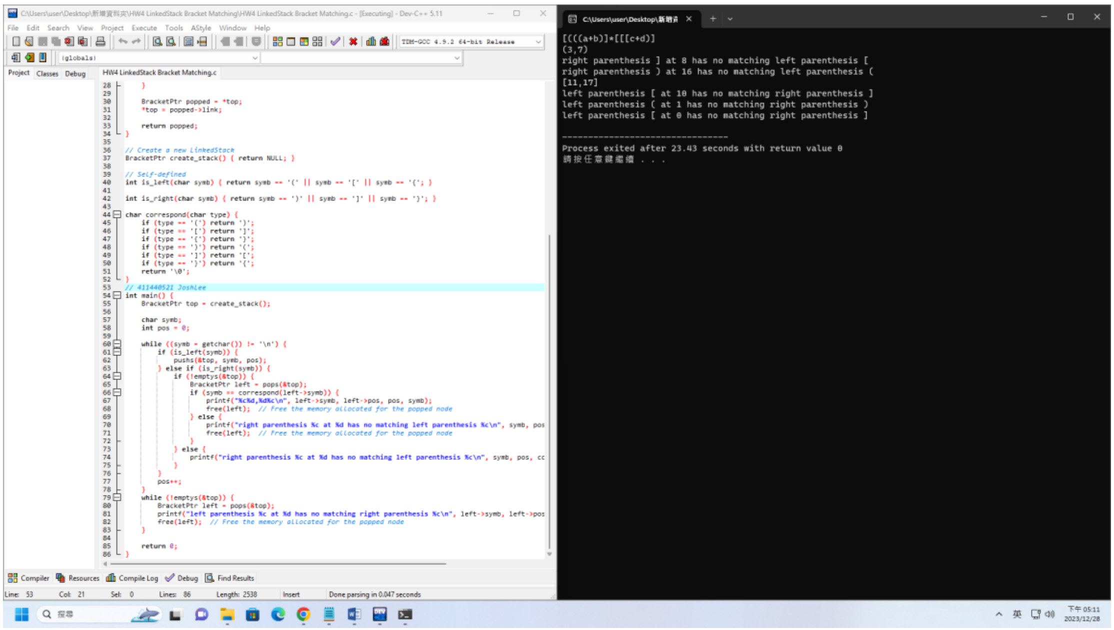

C:\Users\user\Desktop\新增資料夾\HW4 LinkedStack\HW4 LinkedStack.c - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
HW4 LinkedStack.c
Compiler Resources Compile Log Debug Find Results
Line: 60 Col: 21 Sel: 0 Lines: 158 Length: 4802 Insert Done parsing in 0 seconds

```c
                elements_in_current_line = 0;
            }
        }
    printf(elements_in_current_line ? "\n" : "");
}
// 411440521 JoshLee
// Question
void random_number_generator(NodePtr *top, int range, int offset, int len) {
    srand(time(NULL));
    while (!emptys(top)) pops(top);
    int i;
    for (i = 0; i < len; i++) pushs(top, rand() % range + offset);
}

// Helper function to reverse the stack
void reverse_stack(NodePtr *top) {
    NodePtr s1 = create_stack();
    NodePtr s2 = create_stack();

    // Move elements from the original stack to s1
    while (!emptys(top)) {
        NodePtr popped = pops(top);
        pushs(&s1, popped->data);
        free(popped);
    }

    // Move elements from s1 to s2
    while (!emptys(&s1)) {
        NodePtr popped = pops(&s1);
        pushs(&s2, popped->data);
        free(popped);
    }

    // Move elements from s2 back to the original stack
    while (!emptys(&s2)) {
        NodePtr popped = pops(&s2);
        pushs(top, popped->data);
        free(popped);
    }
}

// Helper function to recover elements to the original stack
void recover_elements(NodePtr *top, NodePtr *temp_s) {
    while (!emptys(temp_s)) {
        NodePtr popped = pops(temp_s);
        pushs(top, popped->data);
        free(popped);
    }
}
// 411440521 JoshLee
// Helper function to get an element from the stack
int getElement(NodePtr *top, Direction direction, int position, RecoveryOption recovery) {
    NodePtr temp_s = create_stack();
    int popped_value, i;

    if (direction == FROM_TOP) {
        for (i = 0; i < position; i++) {
            NodePtr popped = pops(top);
            pushs(&temp_s, popped->data);
            popped_value = popped->data;
            free(popped);
```


C:\Users\user\Desktop\新增資料夾\HW4 LinkedStack\HW4 LinkedStack.c - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
HW4 LinkedStack.c
Compiler Resources Compile Log Debug Find Results
Line: 104 Col: 21 Sel: 0 Lines: 158 Length: 4802 Insert Done parsing in 0 seconds

```c
            pushs(top, popped->data);
            free(popped);
        }
}

// Helper function to recover elements to the original stack
void recover_elements(NodePtr *top, NodePtr *temp_s) {
    while (!emptys(temp_s)) {
        NodePtr popped = pops(temp_s);
        pushs(top, popped->data);
        free(popped);
    }
}
// 411440521 JoshLee
// Helper function to get an element from the stack
int getElement(NodePtr *top, Direction direction, int position, RecoveryOption recovery) {
    NodePtr temp_s = create_stack();
    int popped_value, i;

    if (direction == FROM_TOP) {
        for (i = 0; i < position; i++) {
            NodePtr popped = pops(top);
            pushs(&temp_s, popped->data);
            popped_value = popped->data;
            free(popped);
        }
        if (recovery == RECOVER) recover_elements(top, &temp_s);
    } else {
        reverse_stack(top);  // reverse s to pop the bottom n-th element
        for (i = 0; i < position; i++) {
            NodePtr popped = pops(top);
            pushs(&temp_s, popped->data);
            popped_value = popped->data;
            free(popped);
        }
        if (recovery == RECOVER) recover_elements(top, &temp_s);
        reverse_stack(top);  // reverse s to recover_elements to the original direction
    }

    return popped_value;
}
// 411440521 JoshLee
int main() {
    NodePtr s = create_stack();
    int m;

    // 1. Use rand()%100+1 to get 30 random numbers, output the numbers(one by one, one space in between, and 8 numbers in one line)and push the numbers into S one by one
    printf("1. Use rand()%100+1 to get 30 random numbers, output the numbers(one by one, one space in between, and 8 numbers in one line)and push the numbers into S one by one\n");
    random_number_generator(&s, 100, 1, 30);
    prints(s, 8);  // 10 numbers per line
    printf("\n\n");

    // 2. Assign and output integer x the 11th element from the top of  S,leaving S unchanged.
    printf("2. Assign and output integer x the 11th element from the top of  S,leaving S unchanged.\n");
    int x = getElement(&s, FROM_TOP, 11, RECOVER);
    printf("x = %d\n", x);
    prints(s, 8);
    printf("\n\n");

    // 3. Put integer x in (2) under the bottom of S (leaving the rest of  Sunchanged) and output the numbers (one by one, one space inbetween, and 8 numbers in one line) on S from the top to thebottom.
    reverse_stack(&s);
```

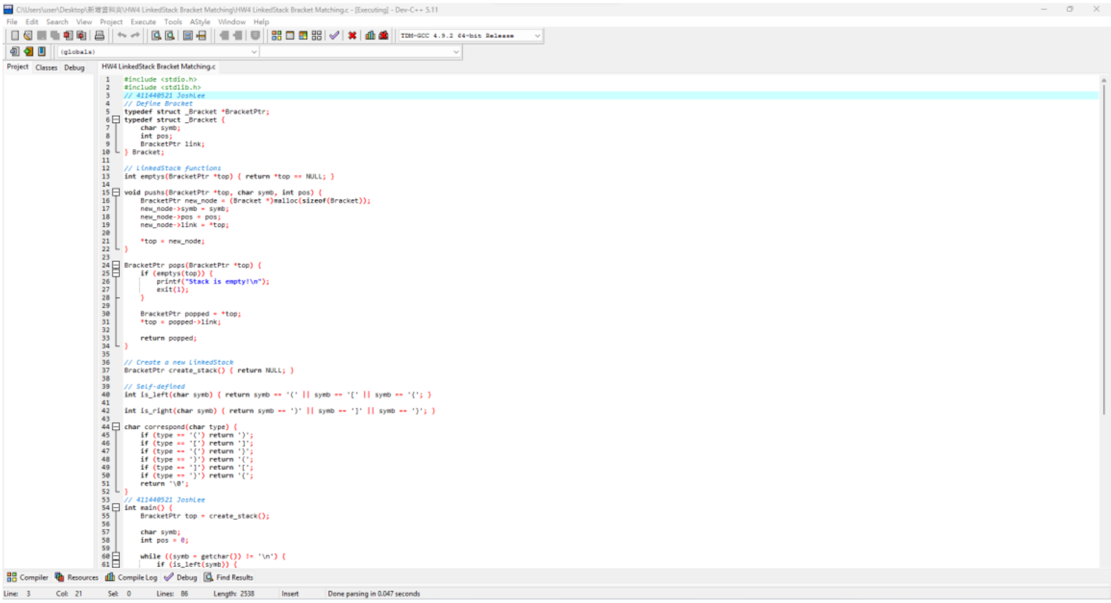
1.4 Bracket Matching



C:\Users\user\Desktop\新增資料夾\HW4 LinkedStack Bracket Matching\HW4 LinkedStack Bracket Matching.c - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
HW4 LinkedStack Bracket Matching.c
Compiler Resources Compile Log Debug Find Results
Line: 53 Col: 21 Sel: 0 Lines: 86 Length: 2538 Insert Done parsing in 0.047 seconds
C:\Users\user\Desktop\新增資
下午 05:11
2023/12/28

```c
    }

    BracketPtr popped = *top;
    *top = popped->link;

    return popped;
}

// Create a new LinkedStack
BracketPtr create_stack() { return NULL; }

// Self-defined
int is_left(char symb) { return symb == '(' || symb == '[' || symb == '{'; }

int is_right(char symb) { return symb == ')' || symb == ']' || symb == '}'; }

char correspond(char type) {
    if (type == '(') return ')';
    if (type == '[') return ']';
    if (type == '{') return '}';
    if (type == ')') return '(';
    if (type == ']') return '[';
    if (type == '}') return '{';
    return '\0';
}
// 411440521 JoshLee
int main() {
    BracketPtr top = create_stack();

    char symb;
    int pos = 0;

    while ((symb = getchar()) != '\n') {
        if (is_left(symb)) {
            pushs(&top, symb, pos);
        } else if (is_right(symb)) {
            if (!emptys(&top)) {
                BracketPtr left = pops(&top);
                if (symb == correspond(left->symb)) {
                    printf("%c%d,%d%c\n", left->symb, left->pos, pos, symb);
                    free(left);  // Free the memory allocated for the popped node
                } else {
                    printf("right parenthesis %c at %d has no matching left parenthesis %c\n", symb, pos
                    free(left);  // Free the memory allocated for the popped node
                }
            } else {
                printf("right parenthesis %c at %d has no matching left parenthesis %c\n", symb, pos, cc
            }
        }
        pos++;
    }
    while (!emptys(&top)) {
        BracketPtr left = pops(&top);
        printf("left parenthesis %c at %d has no matching right parenthesis %c\n", left->symb, left->pos
        free(left);  // Free the memory allocated for the popped node
    }

    return 0;
}
```

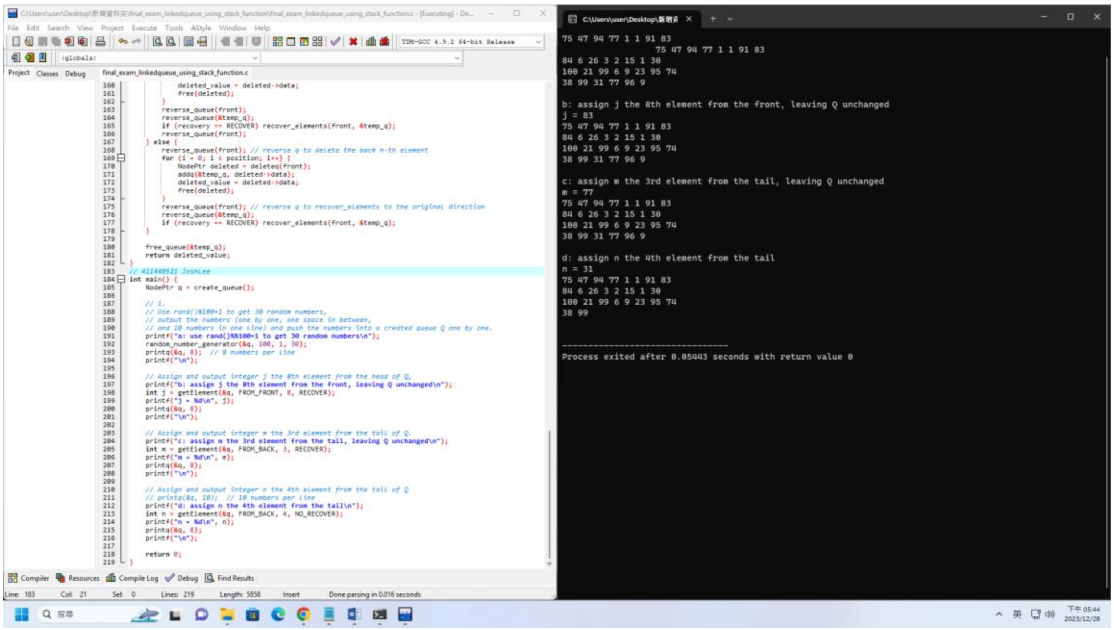
```
[(((a+b)]*[[[c+d)]
(3,7)
right parenthesis ] at 8 has no matching left parenthesis [
right parenthesis ) at 16 has no matching left parenthesis (
[11,17]
left parenthesis [ at 10 has no matching right parenthesis ]
left parenthesis ( at 1 has no matching right parenthesis )
left parenthesis [ at 0 has no matching right parenthesis ]

--------------------------------
Process exited after 23.43 seconds with return value 0
請按任意鍵繼續 . . .
```



C:\Users\user\Desktop\新增資料夾\HW4 LinkedStack Bracket Matching\HW4 LinkedStack Bracket Matching.c - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
HW4 LinkedStack Bracket Matching.c
Compiler Resources Compile Log Debug Find Results
Line: 3 Col: 21 Sel: 0 Lines: 86 Length: 2538 Insert Done parsing in 0.047 seconds

```c
#include <stdio.h>
#include <stdlib.h>
// 411440521 JoshLee
// Define Bracket
typedef struct _Bracket *BracketPtr;
typedef struct _Bracket {
    char symb;
    int pos;
    BracketPtr link;
} Bracket;

// LinkedStack functions
int emptys(BracketPtr *top) { return *top == NULL; }

void pushs(BracketPtr *top, char symb, int pos) {
    BracketPtr new_node = (Bracket *)malloc(sizeof(Bracket));
    new_node->symb = symb;
    new_node->pos = pos;
    new_node->link = *top;

    *top = new_node;
}

BracketPtr pops(BracketPtr *top) {
    if (emptys(top)) {
        printf("Stack is empty!\n");
        exit(1);
    }

    BracketPtr popped = *top;
    *top = popped->link;

    return popped;
}

// Create a new LinkedStack
BracketPtr create_stack() { return NULL; }

// Self-defined
int is_left(char symb) { return symb == '(' || symb == '[' || symb == '{'; }

int is_right(char symb) { return symb == ')' || symb == ']' || symb == '}'; }

char correspond(char type) {
    if (type == '(') return ')';
    if (type == '[') return ']';
    if (type == '{') return '}';
    if (type == ')') return '(';
    if (type == ']') return '[';
    if (type == '}') return '{';
    return '\0';
}
// 411440521 JoshLee
int main() {
    BracketPtr top = create_stack();

    char symb;
    int pos = 0;

    while ((symb = getchar()) != '\n') {
        if (is_left(symb)) {
```

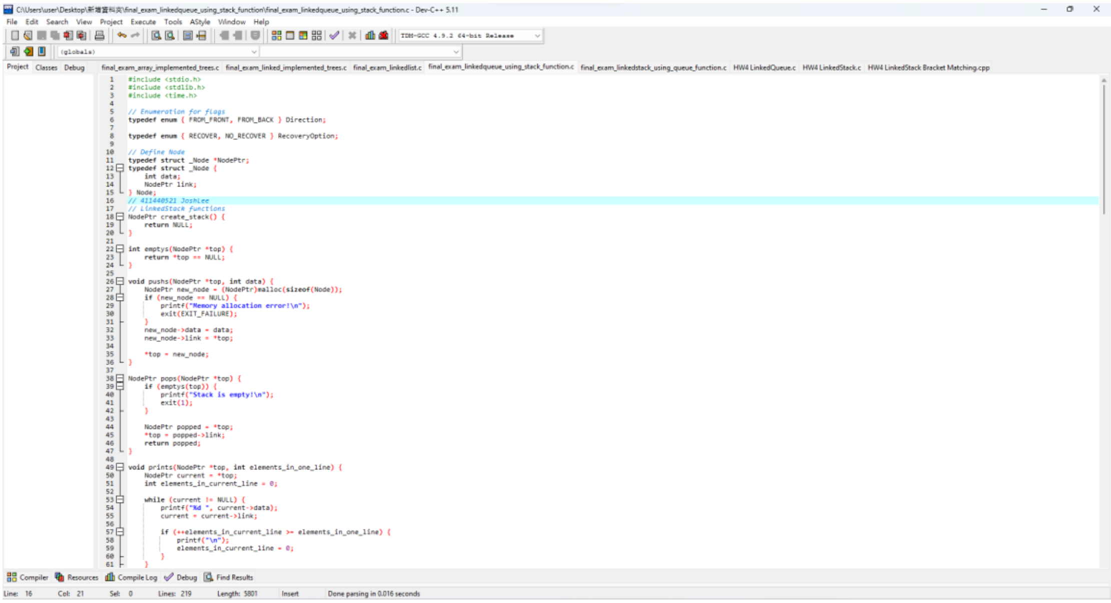
2.1 Linked Queues by Linked Stacks



```c
            deleted_value = deleted->data;
            free(deleted);
        }
        reverse_queue(front);
        reverse_queue(&temp_q);
        if (recovery == RECOVER) recover_elements(front, &temp_q);
        reverse_queue(front);
    } else {
        reverse_queue(front); // reverse q to delete the back n-th element
        for (i = 0; i < position; i++) {
            NodePtr deleted = deleteq(front);
            addq(&temp_q, deleted->data);
            deleted_value = deleted->data;
            free(deleted);
        }
        reverse_queue(front); // reverse q to recover_elements to the original direction
        reverse_queue(&temp_q);
        if (recovery == RECOVER) recover_elements(front, &temp_q);
    }

    free_queue(&temp_q);
    return deleted_value;
}
// 411440521 JoshLee
int main() {
    NodePtr q = create_queue();

    // 1.
    // Use rand()%100+1 to get 30 random numbers,
    // output the numbers (one by one, one space in between,
    // and 10 numbers in one line) and push the numbers into a created queue Q one by one.
    printf("a: use rand()%%100+1 to get 30 random numbers\n");
    random_number_generator(&q, 100, 1, 30);
    printq(&q, 8);  // 8 numbers per line
    printf("\n");

    // Assign and output integer j the 8th element from the head of Q,
    printf("b: assign j the 8th element from the front, leaving Q unchanged\n");
    int j = getElement(&q, FROM_FRONT, 8, RECOVER);
    printf("j = %d\n", j);
    printq(&q, 8);
    printf("\n");

    // Assign and output integer m the 3rd element from the tail of Q.
    printf("c: assign m the 3rd element from the tail, leaving Q unchanged\n");
    int m = getElement(&q, FROM_BACK, 3, RECOVER);
    printf("m = %d\n", m);
    printq(&q, 8);
    printf("\n");

    // Assign and output integer n the 4th element from the tail of Q
    // printq(&q, 10);  // 10 numbers per line
    printf("d: assign n the 4th element from the tail\n");
    int n = getElement(&q, FROM_BACK, 4, NO_RECOVER);
    printf("n = %d\n", n);
    printq(&q, 8);
    printf("\n");

    return 0;
}
```

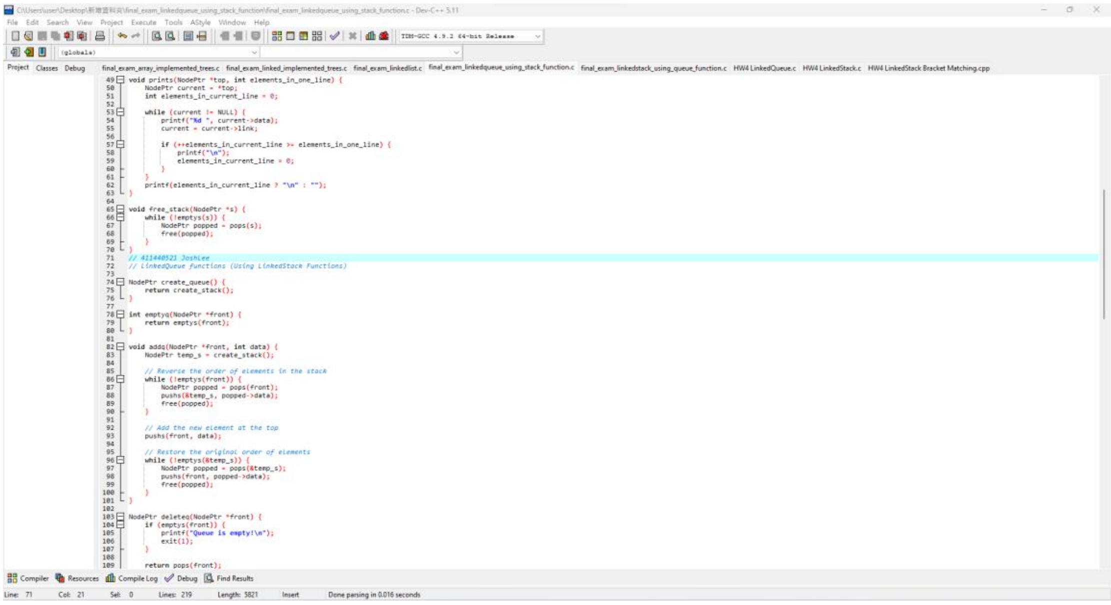
```
75 47 94 77 1 1 91 83
                  75 47 94 77 1 1 91 83
84 6 26 3 2 15 1 30
100 21 99 6 9 23 95 74
38 99 31 77 96 9

b: assign j the 8th element from the front, leaving Q unchanged
j = 83
75 47 94 77 1 1 91 83
84 6 26 3 2 15 1 30
100 21 99 6 9 23 95 74
38 99 31 77 96 9

c: assign m the 3rd element from the tail, leaving Q unchanged
m = 77
75 47 94 77 1 1 91 83
84 6 26 3 2 15 1 30
100 21 99 6 9 23 95 74
38 99 31 77 96 9

d: assign n the 4th element from the tail
n = 31
75 47 94 77 1 1 91 83
84 6 26 3 2 15 1 30
100 21 99 6 9 23 95 74
38 99


--------------------------------
Process exited after 0.05443 seconds with return value 0
```

C:\Users\user\Desktop\新增資料夾\final_exam_linkedqueue_using_stack_function\final_exam_linkedqueue_using_stack_function.c - [Executing] - De...
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_linkedqueue_using_stack_function.c
Compiler Resources Compile Log Debug Find Results
Line: 183 Col: 21 Sel: 0 Lines: 219 Length: 5858 Insert Done parsing in 0.016 seconds
C:\Users\user\Desktop\新增資
下午 05:44
2023/12/28



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

// Enumeration for flags
typedef enum { FROM_FRONT, FROM_BACK } Direction;

typedef enum { RECOVER, NO_RECOVER } RecoveryOption;

// Define Node
typedef struct _Node *NodePtr;
typedef struct _Node {
    int data;
    NodePtr link;
} Node;
// 411440521 JoshLee
// LinkedStack functions
NodePtr create_stack() {
    return NULL;
}

int emptys(NodePtr *top) {
    return *top == NULL;
}

void pushs(NodePtr *top, int data) {
    NodePtr new_node = (NodePtr)malloc(sizeof(Node));
    if (new_node == NULL) {
        printf("Memory allocation error!\n");
        exit(EXIT_FAILURE);
    }
    new_node->data = data;
    new_node->link = *top;

    *top = new_node;
}

NodePtr pops(NodePtr *top) {
    if (emptys(top)) {
        printf("Stack is empty!\n");
        exit(1);
    }

    NodePtr popped = *top;
    *top = popped->link;
    return popped;
}

void prints(NodePtr *top, int elements_in_one_line) {
    NodePtr current = *top;
    int elements_in_current_line = 0;

    while (current != NULL) {
        printf("%d ", current->data);
        current = current->link;

        if (++elements_in_current_line >= elements_in_one_line) {
            printf("\n");
            elements_in_current_line = 0;
        }
    }
```

C:\Users\user\Desktop\新增資料夾\final_exam_linkedqueue_using_stack_function\final_exam_linkedqueue_using_stack_function.c - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_array_implemented_trees.c
final_exam_linked_implemented_trees.c
final_exam_linkedlist.c
final_exam_linkedqueue_using_stack_function.c
final_exam_linkedstack_using_queue_function.c
HW4 LinkedQueue.c
HW4 LinkedStack.c
HW4 LinkedStack Bracket Matching.cpp
Compiler Resources Compile Log Debug Find Results
Line: 16 Col: 21 Sel: 0 Lines: 219 Length: 5801 Insert Done parsing in 0.016 seconds

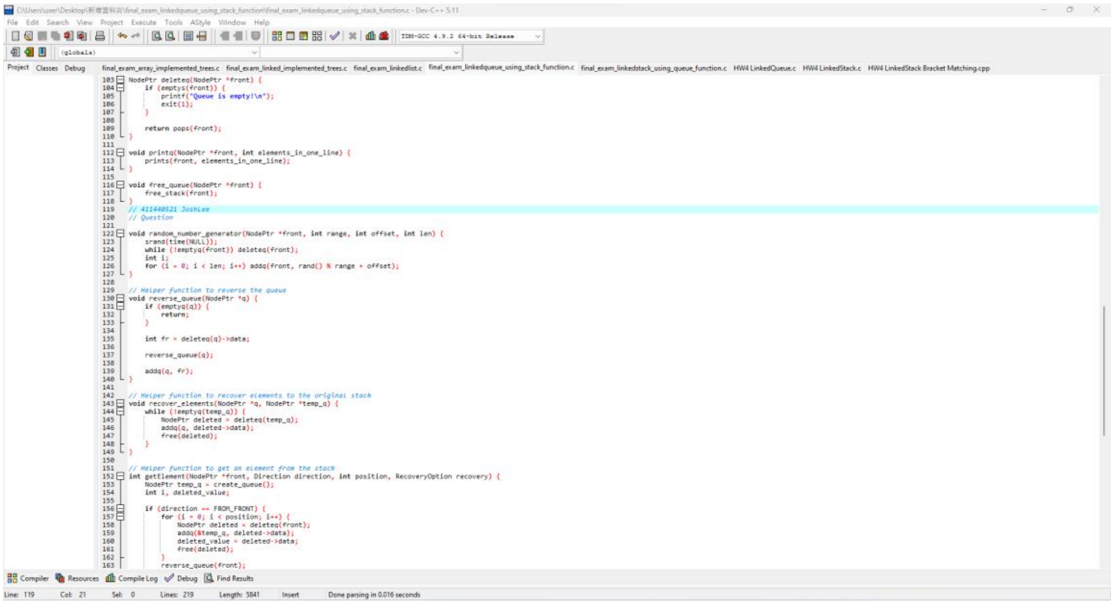


C:\Users\user\Desktop\新增資料夾\final_exam_linkedqueue_using_stack_function\final_exam_linkedqueue_using_stack_function.c - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_array_implemented_trees.c final_exam_linked_implemented_trees.c final_exam_linkedlist.c final_exam_linkedqueue_using_stack_function.c final_exam_linkedstack_using_queue_function.c HW4 LinkedQueue.c HW4 LinkedStack.c HW4 LinkedStack Bracket Matching.cpp
Compiler Resources Compile Log Debug Find Results

```c
void prints(NodePtr *top, int elements_in_one_line) {
    NodePtr current = *top;
    int elements_in_current_line = 0;

    while (current != NULL) {
        printf("%d ", current->data);
        current = current->link;

        if (++elements_in_current_line >= elements_in_one_line) {
            printf("\n");
            elements_in_current_line = 0;
        }
    }
    printf(elements_in_current_line ? "\n" : "");
}

void free_stack(NodePtr *s) {
    while (!emptys(s)) {
        NodePtr popped = pops(s);
        free(popped);
    }
}
// 411440521 JoshLee
// LinkedQueue functions (Using LinkedStack Functions)

NodePtr create_queue() {
    return create_stack();
}

int emptyq(NodePtr *front) {
    return emptys(front);
}

void addq(NodePtr *front, int data) {
    NodePtr temp_s = create_stack();

    // Reverse the order of elements in the stack
    while (!emptys(front)) {
        NodePtr popped = pops(front);
        pushs(&temp_s, popped->data);
        free(popped);
    }

    // Add the new element at the top
    pushs(front, data);

    // Restore the original order of elements
    while (!emptys(&temp_s)) {
        NodePtr popped = pops(&temp_s);
        pushs(front, popped->data);
        free(popped);
    }
}

NodePtr deleteq(NodePtr *front) {
    if (emptys(front)) {
        printf("Queue is empty!\n");
        exit(1);
    }

    return pops(front);
```

Line: 71 Col: 21 Sel: 0 Lines: 219 Length: 5821 Insert Done parsing in 0.016 seconds

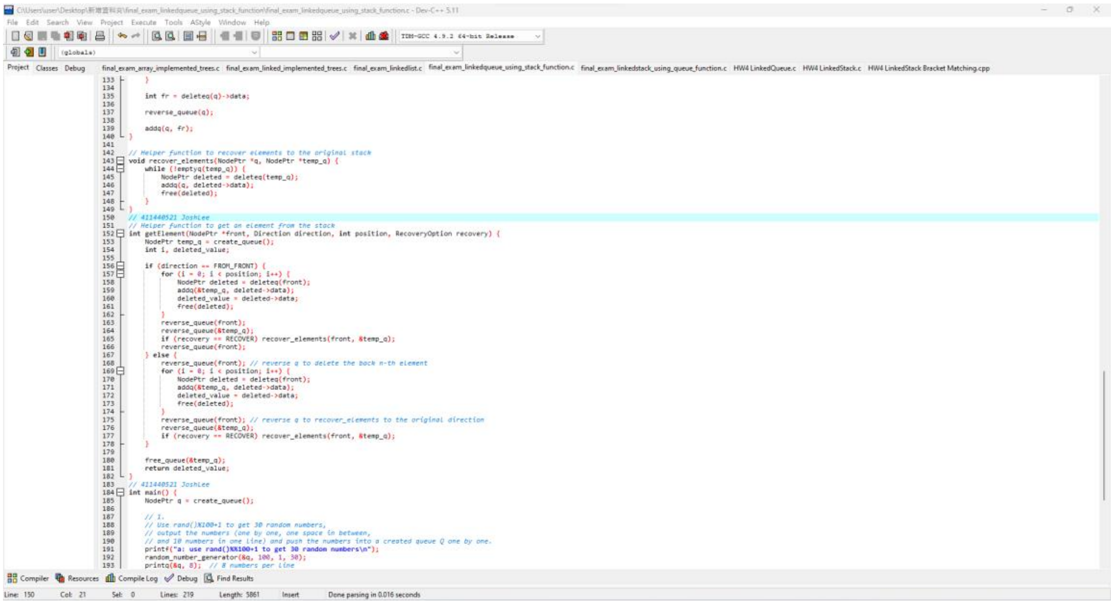


C:\Users\user\Desktop\新增資料夾\final_exam_linkedqueue_using_stack_function\final_exam_linkedqueue_using_stack_function.c - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_array_implemented_trees.c final_exam_linked_implemented_trees.c final_exam_linkedlist.c final_exam_linkedqueue_using_stack_function.c final_exam_linkedstack_using_queue_function.c HW4 LinkedQueue.c HW4 LinkedStack.c HW4 LinkedStack Bracket Matching.cpp
Compiler Resources Compile Log Debug Find Results

```c
NodePtr deleteq(NodePtr *front) {
    if (emptys(front)) {
        printf("Queue is empty!\n");
        exit(1);
    }

    return pops(front);
}

void printq(NodePtr *front, int elements_in_one_line) {
    prints(front, elements_in_one_line);
}

void free_queue(NodePtr *front) {
    free_stack(front);
}
// 411440521 JoshLee
// Question

void random_number_generator(NodePtr *front, int range, int offset, int len) {
    srand(time(NULL));
    while (!emptyq(front)) deleteq(front);
    int i;
    for (i = 0; i < len; i++) addq(front, rand() % range + offset);
}

// Helper function to reverse the queue
void reverse_queue(NodePtr *q) {
    if (emptyq(q)) {
        return;
    }

    int fr = deleteq(q)->data;

    reverse_queue(q);

    addq(q, fr);
}

// Helper function to recover elements to the original stack
void recover_elements(NodePtr *q, NodePtr *temp_q) {
    while (!emptyq(temp_q)) {
        NodePtr deleted = deleteq(temp_q);
        addq(q, deleted->data);
        free(deleted);
    }
}

// Helper function to get an element from the stack
int getElement(NodePtr *front, Direction direction, int position, RecoveryOption recovery) {
    NodePtr temp_q = create_queue();
    int i, deleted_value;

    if (direction == FROM_FRONT) {
        for (i = 0; i < position; i++) {
            NodePtr deleted = deleteq(front);
            addq(&temp_q, deleted->data);
            deleted_value = deleted->data;
            free(deleted);
        }
        reverse_queue(front);
```

Line: 119 Col: 21 Sel: 0 Lines: 219 Length: 5841 Insert Done parsing in 0.016 seconds



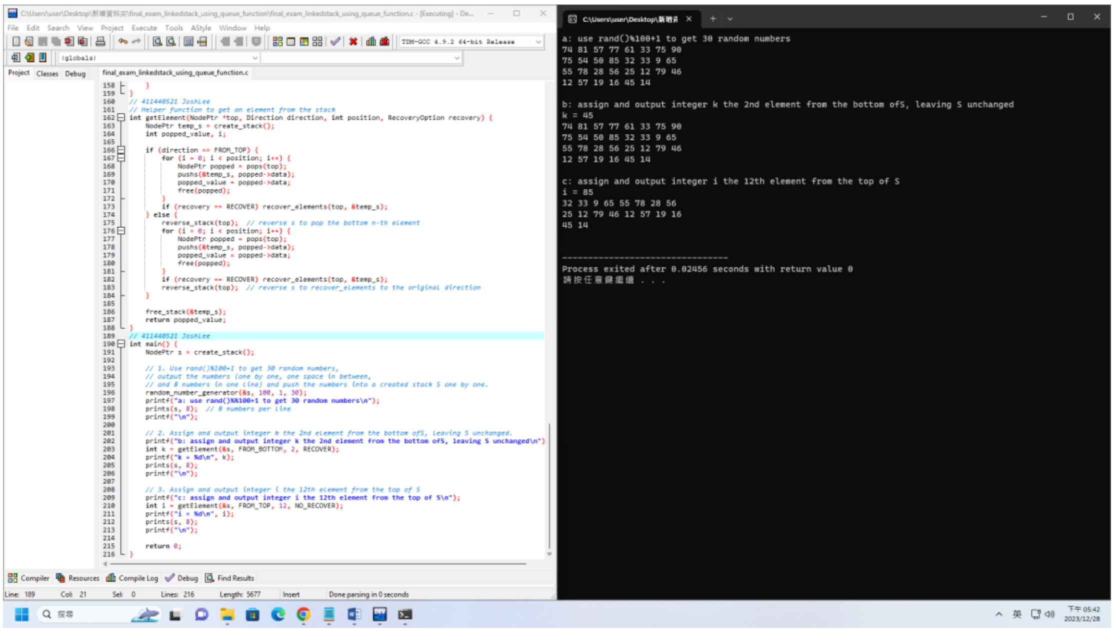
C:\Users\user\Desktop\新增資料夾\final_exam_linkedqueue_using_stack_function\final_exam_linkedqueue_using_stack_function.c - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_array_implemented_trees.c final_exam_linked_implemented_trees.c final_exam_linkedlist.c final_exam_linkedqueue_using_stack_function.c final_exam_linkedstack_using_queue_function.c HW4 LinkedQueue.c HW4 LinkedStack.c HW4 LinkedStack Bracket Matching.cpp
Compiler Resources Compile Log Debug Find Results

```c
    }

    int fr = deleteq(q)->data;

    reverse_queue(q);

    addq(q, fr);
}

// Helper function to recover elements to the original stack
void recover_elements(NodePtr *q, NodePtr *temp_q) {
    while (!emptyq(temp_q)) {
        NodePtr deleted = deleteq(temp_q);
        addq(q, deleted->data);
        free(deleted);
    }
}
// 411440521 JoshLee
// Helper function to get an element from the stack
int getElement(NodePtr *front, Direction direction, int position, RecoveryOption recovery) {
    NodePtr temp_q = create_queue();
    int i, deleted_value;

    if (direction == FROM_FRONT) {
        for (i = 0; i < position; i++) {
            NodePtr deleted = deleteq(front);
            addq(&temp_q, deleted->data);
            deleted_value = deleted->data;
            free(deleted);
        }
        reverse_queue(front);
        reverse_queue(&temp_q);
        if (recovery == RECOVER) recover_elements(front, &temp_q);
        reverse_queue(front);
    } else {
        reverse_queue(front); // reverse q to delete the back n-th element
        for (i = 0; i < position; i++) {
            NodePtr deleted = deleteq(front);
            addq(&temp_q, deleted->data);
            deleted_value = deleted->data;
            free(deleted);
        }
        reverse_queue(front); // reverse q to recover_elements to the original direction
        reverse_queue(&temp_q);
        if (recovery == RECOVER) recover_elements(front, &temp_q);
    }

    free_queue(&temp_q);
    return deleted_value;
}
// 411440521 JoshLee
int main() {
    NodePtr q = create_queue();

    // 1.
    // Use rand()%100+1 to get 30 random numbers,
    // output the numbers (one by one, one space in between,
    // and 10 numbers in one line) and push the numbers into a created queue Q one by one.
    printf("a: use rand()%%100+1 to get 30 random numbers\n");
    random_number_generator(&q, 100, 1, 30);
    printq(&q, 8);  // 8 numbers per line
```

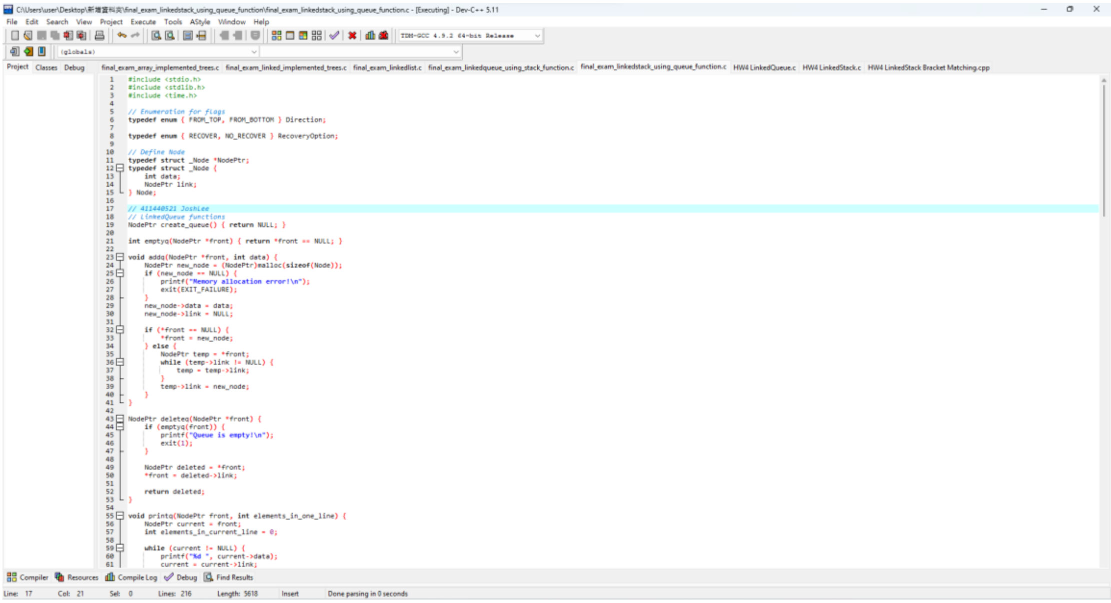
Line: 150 Col: 21 Sel: 0 Lines: 219 Length: 5861 Insert Done parsing in 0.016 seconds

2.2 Linked Stacks by Linked Queues

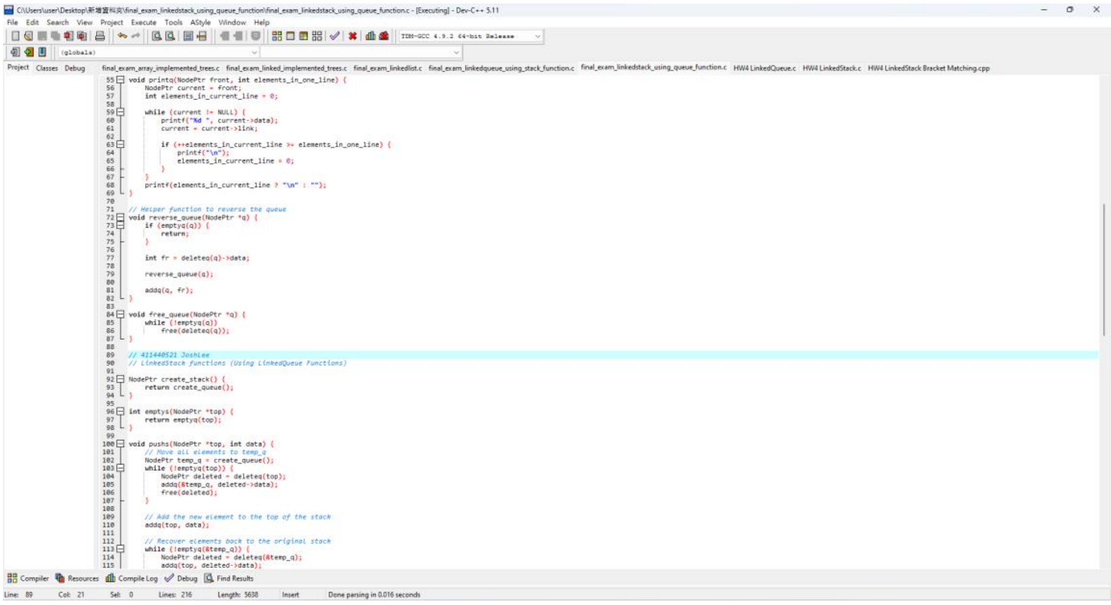

C:\Users\user\Desktop\新增資料夾\final_exam_linkedstack_using_queue_function\final_exam_linkedstack_using_queue_function.c - [Executing] - De...
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_linkedstack_using_queue_function.c
Compiler Resources Compile Log Debug Find Results
Line: 189 Col: 21 Sel: 0 Lines: 216 Length: 5677 Insert Done parsing in 0 seconds
C:\Users\user\Desktop\新增資
下午 05:42
2023/12/28

```c
    }
}
// 411440521 JoshLee
// Helper function to get an element from the stack
int getElement(NodePtr *top, Direction direction, int position, RecoveryOption recovery) {
    NodePtr temp_s = create_stack();
    int popped_value, i;

    if (direction == FROM_TOP) {
        for (i = 0; i < position; i++) {
            NodePtr popped = pops(top);
            pushs(&temp_s, popped->data);
            popped_value = popped->data;
            free(popped);
        }
        if (recovery == RECOVER) recover_elements(top, &temp_s);
    } else {
        reverse_stack(top);  // reverse s to pop the bottom n-th element
        for (i = 0; i < position; i++) {
            NodePtr popped = pops(top);
            pushs(&temp_s, popped->data);
            popped_value = popped->data;
            free(popped);
        }
        if (recovery == RECOVER) recover_elements(top, &temp_s);
        reverse_stack(top);  // reverse s to recover_elements to the original direction
    }

    free_stack(&temp_s);
    return popped_value;
}
// 411440521 JoshLee
int main() {
    NodePtr s = create_stack();

    // 1. Use rand()%100+1 to get 30 random numbers,
    // output the numbers (one by one, one space in between,
    // and 8 numbers in one line) and push the numbers into a created stack S one by one.
    random_number_generator(&s, 100, 1, 30);
    printf("a: use rand()%%100+1 to get 30 random numbers\n");
    prints(s, 8);  // 8 numbers per line
    printf("\n");

    // 2. Assign and output integer k the 2nd element from the bottom ofS, leaving S unchanged.
    printf("b: assign and output integer k the 2nd element from the bottom ofS, leaving S unchanged\n")
    int k = getElement(&s, FROM_BOTTOM, 2, RECOVER);
    printf("k = %d\n", k);
    prints(s, 8);
    printf("\n");

    // 3. Assign and output integer i the 12th element from the top of S
    printf("c: assign and output integer i the 12th element from the top of S\n");
    int i = getElement(&s, FROM_TOP, 12, NO_RECOVER);
    printf("i = %d\n", i);
    prints(s, 8);
    printf("\n");

    return 0;
}
```

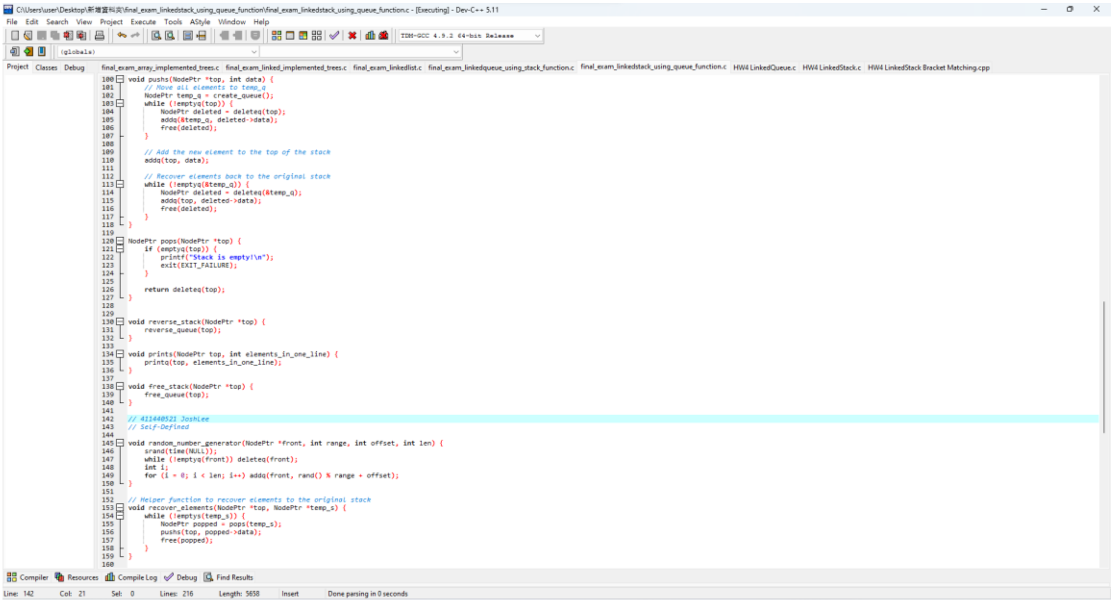
```
a: use rand()%100+1 to get 30 random numbers
74 81 57 77 61 33 75 90
75 54 50 85 32 33 9 65
55 78 28 56 25 12 79 46
12 57 19 16 45 14

b: assign and output integer k the 2nd element from the bottom ofS, leaving S unchanged
k = 45
74 81 57 77 61 33 75 90
75 54 50 85 32 33 9 65
55 78 28 56 25 12 79 46
12 57 19 16 45 14

c: assign and output integer i the 12th element from the top of S
i = 85
32 33 9 65 55 78 28 56
25 12 79 46 12 57 19 16
45 14


--------------------------------
Process exited after 0.02456 seconds with return value 0
請按任意鍵繼續 . . .
```


C:\Users\user\Desktop\新增資料夾\final_exam_linkedstack_using_queue_function\final_exam_linkedstack_using_queue_function.c - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_array_implemented_trees.c
final_exam_linked_implemented_trees.c
final_exam_linkedlist.c
final_exam_linkedqueue_using_stack_function.c
final_exam_linkedstack_using_queue_function.c
HW4 LinkedQueue.c
HW4 LinkedStack.c
HW4 LinkedStack Bracket Matching.cpp
Compiler Resources Compile Log Debug Find Results
Line: 17 Col: 21 Sel: 0 Lines: 216 Length: 5618 Insert Done parsing in 0 seconds

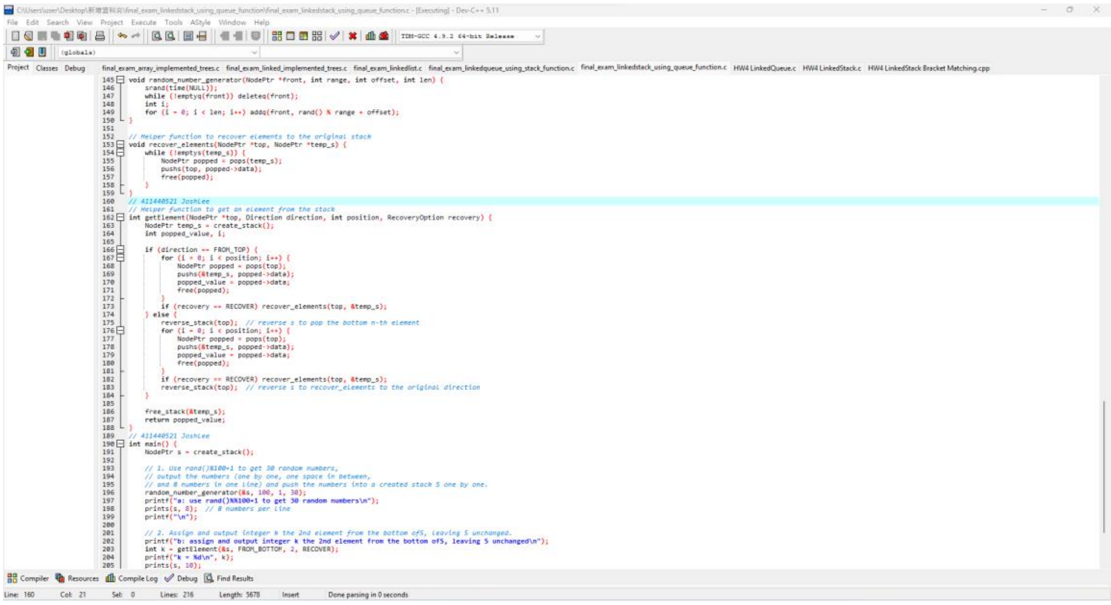
```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

// Enumeration for flags
typedef enum { FROM_TOP, FROM_BOTTOM } Direction;

typedef enum { RECOVER, NO_RECOVER } RecoveryOption;

// Define Node
typedef struct _Node *NodePtr;
typedef struct _Node {
    int data;
    NodePtr link;
} Node;

// 411440521 JoshLee
// LinkedQueue functions
NodePtr create_queue() { return NULL; }

int emptyq(NodePtr *front) { return *front == NULL; }

void addq(NodePtr *front, int data) {
    NodePtr new_node = (NodePtr)malloc(sizeof(Node));
    if (new_node == NULL) {
        printf("Memory allocation error!\n");
        exit(EXIT_FAILURE);
    }
    new_node->data = data;
    new_node->link = NULL;

    if (*front == NULL) {
        *front = new_node;
    } else {
        NodePtr temp = *front;
        while (temp->link != NULL) {
            temp = temp->link;
        }
        temp->link = new_node;
    }
}

NodePtr deleteq(NodePtr *front) {
    if (emptyq(front)) {
        printf("Queue is empty!\n");
        exit(1);
    }

    NodePtr deleted = *front;
    *front = deleted->link;

    return deleted;
}

void printq(NodePtr front, int elements_in_one_line) {
    NodePtr current = front;
    int elements_in_current_line = 0;

    while (current != NULL) {
        printf("%d ", current->data);
        current = current->link;
```


C:\Users\user\Desktop\新增資料夾\final_exam_linkedstack_using_queue_function\final_exam_linkedstack_using_queue_function.c - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_array_implemented_trees.c final_exam_linked_implemented_trees.c final_exam_linkedlist.c final_exam_linkedqueue_using_stack_function.c final_exam_linkedstack_using_queue_function.c HW4 LinkedQueue.c HW4 LinkedStack.c HW4 LinkedStack Bracket Matching.cpp
Compiler Resources Compile Log Debug Find Results
Line: 89 Col: 21 Sel: 0 Lines: 216 Length: 5638 Insert Done parsing in 0.016 seconds

```c
void printq(NodePtr front, int elements_in_one_line) {
    NodePtr current = front;
    int elements_in_current_line = 0;

    while (current != NULL) {
        printf("%d ", current->data);
        current = current->link;

        if (++elements_in_current_line >= elements_in_one_line) {
            printf("\n");
            elements_in_current_line = 0;
        }
    }
    printf(elements_in_current_line ? "\n" : "");
}

// Helper function to reverse the queue
void reverse_queue(NodePtr *q) {
    if (emptyq(q)) {
        return;
    }

    int fr = deleteq(q)->data;

    reverse_queue(q);

    addq(q, fr);
}

void free_queue(NodePtr *q) {
    while (!emptyq(q))
        free(deleteq(q));
}

// 411440521 JoshLee
// LinkedStack functions (Using LinkedQueue Functions)

NodePtr create_stack() {
    return create_queue();
}

int emptys(NodePtr *top) {
    return emptyq(top);
}

void pushs(NodePtr *top, int data) {
    // Move all elements to temp_q
    NodePtr temp_q = create_queue();
    while (!emptyq(top)) {
        NodePtr deleted = deleteq(top);
        addq(&temp_q, deleted->data);
        free(deleted);
    }

    // Add the new element to the top of the stack
    addq(top, data);

    // Recover elements back to the original stack
    while (!emptyq(&temp_q)) {
        NodePtr deleted = deleteq(&temp_q);
        addq(top, deleted->data);
```

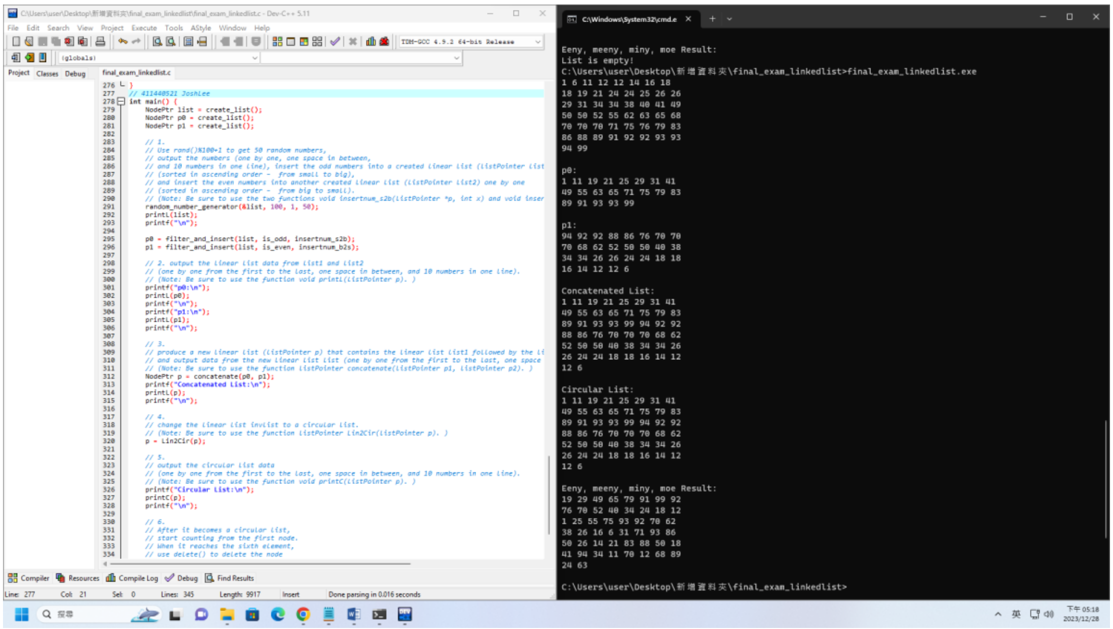

C:\Users\user\Desktop\新增資料夾\final_exam_linkedstack_using_queue_function\final_exam_linkedstack_using_queue_function.c - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_array_implemented_trees.c final_exam_linked_implemented_trees.c final_exam_linkedlist.c final_exam_linkedqueue_using_stack_function.c final_exam_linkedstack_using_queue_function.c HW4 LinkedQueue.c HW4 LinkedStack.c HW4 LinkedStack Bracket Matching.cpp
Compiler Resources Compile Log Debug Find Results
Line: 142 Col: 21 Sel: 0 Lines: 216 Length: 5658 Insert Done parsing in 0 seconds

```c
void pushs(NodePtr *top, int data) {
    // Move all elements to temp_q
    NodePtr temp_q = create_queue();
    while (!emptyq(top)) {
        NodePtr deleted = deleteq(top);
        addq(&temp_q, deleted->data);
        free(deleted);
    }

    // Add the new element to the top of the stack
    addq(top, data);

    // Recover elements back to the original stack
    while (!emptyq(&temp_q)) {
        NodePtr deleted = deleteq(&temp_q);
        addq(top, deleted->data);
        free(deleted);
    }
}

NodePtr pops(NodePtr *top) {
    if (emptyq(top)) {
        printf("Stack is empty!\n");
        exit(EXIT_FAILURE);
    }

    return deleteq(top);
}


void reverse_stack(NodePtr *top) {
    reverse_queue(top);
}

void prints(NodePtr top, int elements_in_one_line) {
    printq(top, elements_in_one_line);
}

void free_stack(NodePtr *top) {
    free_queue(top);
}

// 411440521 JoshLee
// Self-Defined

void random_number_generator(NodePtr *front, int range, int offset, int len) {
    srand(time(NULL));
    while (!emptyq(front)) deleteq(front);
    int i;
    for (i = 0; i < len; i++) addq(front, rand() % range + offset);
}

// Helper function to recover elements to the original stack
void recover_elements(NodePtr *top, NodePtr *temp_s) {
    while (!emptys(temp_s)) {
        NodePtr popped = pops(temp_s);
        pushs(top, popped->data);
        free(popped);
    }
}

```


C:\Users\user\Desktop\新增資料夾\final_exam_linkedstack_using_queue_function\final_exam_linkedstack_using_queue_function.c - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_array_implemented_trees.c final_exam_linked_implemented_trees.c final_exam_linkedlist.c final_exam_linkedqueue_using_stack_function.c final_exam_linkedstack_using_queue_function.c HW4 LinkedQueue.c HW4 LinkedStack.c HW4 LinkedStack Bracket Matching.cpp
Compiler Resources Compile Log Debug Find Results
Line: 160 Col: 21 Sel: 0 Lines: 216 Length: 5678 Insert Done parsing in 0 seconds

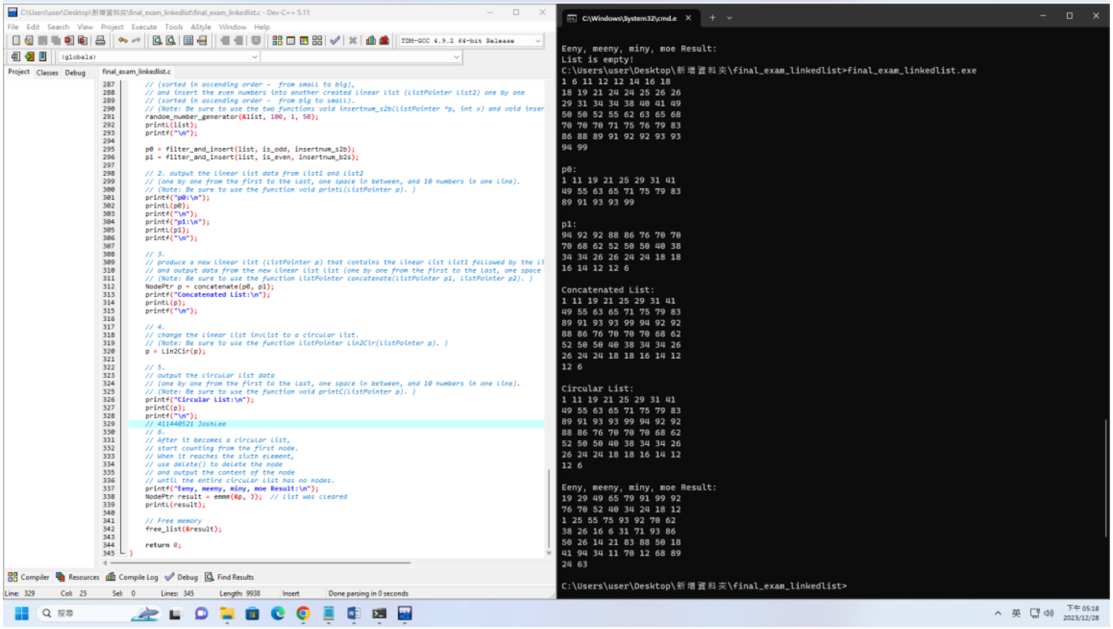
```c
void random_number_generator(NodePtr *front, int range, int offset, int len) {
    srand(time(NULL));
    while (!emptyq(front)) deleteq(front);
    int i;
    for (i = 0; i < len; i++) addq(front, rand() % range + offset);
}

// Helper function to recover elements to the original stack
void recover_elements(NodePtr *top, NodePtr *temp_s) {
    while (!emptys(temp_s)) {
        NodePtr popped = pops(temp_s);
        pushs(top, popped->data);
        free(popped);
    }
}
// 411440521 JoshLee
// Helper function to get an element from the stack
int getElement(NodePtr *top, Direction direction, int position, RecoveryOption recovery) {
    NodePtr temp_s = create_stack();
    int popped_value, i;

    if (direction == FROM_TOP) {
        for (i = 0; i < position; i++) {
            NodePtr popped = pops(top);
            pushs(&temp_s, popped->data);
            popped_value = popped->data;
            free(popped);
        }
        if (recovery == RECOVER) recover_elements(top, &temp_s);
    } else {
        reverse_stack(top);  // reverse s to pop the bottom n-th element
        for (i = 0; i < position; i++) {
            NodePtr popped = pops(top);
            pushs(&temp_s, popped->data);
            popped_value = popped->data;
            free(popped);
        }
        if (recovery == RECOVER) recover_elements(top, &temp_s);
        reverse_stack(top);  // reverse s to recover_elements to the original direction
    }

    free_stack(&temp_s);
    return popped_value;
}
// 411440521 JoshLee
int main() {
    NodePtr s = create_stack();

    // 1. Use rand()%100+1 to get 30 random numbers,
    // output the numbers (one by one, one space in between,
    // and 8 numbers in one line) and push the numbers into a created stack S one by one.
    random_number_generator(&s, 100, 1, 30);
    printf("a: use rand()%%100+1 to get 30 random numbers\n");
    prints(s, 8);  // 8 numbers per line
    printf("\n");

    // 2. Assign and output integer k the 2nd element from the bottom ofS, leaving S unchanged.
    printf("b: assign and output integer k the 2nd element from the bottom ofS, leaving S unchanged\n");
    int k = getElement(&s, FROM_BOTTOM, 2, RECOVER);
    printf("k = %d\n", k);
    prints(s, 10);
```

3. Linked List


C:\Users\user\Desktop\新增資料夾\final_exam_linkedlist\final_exam_linkedlist.c - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_linkedlist.c
Compiler Resources Compile Log Debug Find Results
Line: 277 Col: 21 Sel: 0 Lines: 345 Length: 9917 Insert Done parsing in 0.016 seconds
C:\Windows\System32\cmd.e

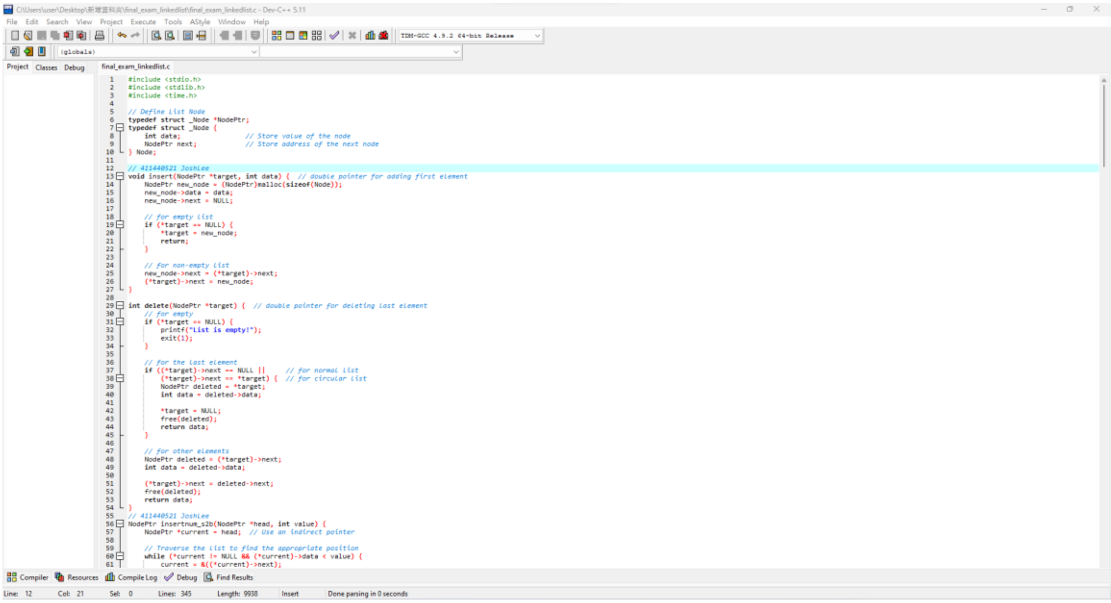
```c
}
// 411440521 JoshLee
int main() {
    NodePtr list = create_list();
    NodePtr p0 = create_list();
    NodePtr p1 = create_list();

    // 1.
    // Use rand()%100+1 to get 50 random numbers,
    // output the numbers (one by one, one space in between,
    // and 10 numbers in one line), insert the odd numbers into a created linear list (listPointer list
    // (sorted in ascending order -  from small to big),
    // and insert the even numbers into another created linear list (listPointer list2) one by one
    // (sorted in ascending order -  from big to small).
    // (Note: Be sure to use the two functions void insertnum_s2b(listPointer *p, int x) and void inser
    random_number_generator(&list, 100, 1, 50);
    printL(list);
    printf("\n");

    p0 = filter_and_insert(list, is_odd, insertnum_s2b);
    p1 = filter_and_insert(list, is_even, insertnum_b2s);

    // 2. output the linear list data from list1 and list2
    // (one by one from the first to the last, one space in between, and 10 numbers in one line).
    // (Note: Be sure to use the function void printL(listPointer p). )
    printf("p0:\n");
    printL(p0);
    printf("\n");
    printf("p1:\n");
    printL(p1);
    printf("\n");

    // 3.
    // produce a new linear list (listPointer p) that contains the linear list list1 followed by the li
    // and output data from the new linear list list (one by one from the first to the last, one space
    // (Note: Be sure to use the function listPointer concatenate(listPointer p1, listPointer p2). )
    NodePtr p = concatenate(p0, p1);
    printf("Concatenated List:\n");
    printL(p);
    printf("\n");

    // 4.
    // change the linear list invlist to a circular list.
    // (Note: Be sure to use the function listPointer Lin2Cir(listPointer p). )
    p = Lin2Cir(p);

    // 5.
    // output the circular list data
    // (one by one from the first to the last, one space in between, and 10 numbers in one line).
    // (Note: Be sure to use the function void printC(listPointer p). )
    printf("Circular List:\n");
    printC(p);
    printf("\n");

    // 6.
    // After it becomes a circular list,
    // start counting from the first node.
    // When it reaches the sixth element,
    // use delete() to delete the node
```

```
Eeny, meeny, miny, moe Result:
List is empty!
C:\Users\user\Desktop\新增資料夾\final_exam_linkedlist>final_exam_linkedlist.exe
1 6 11 12 12 14 16 18
18 19 21 24 24 25 26 26
29 31 34 34 38 40 41 49
50 50 52 55 62 63 65 68
70 70 70 71 75 76 79 83
86 88 89 91 92 92 93 93
94 99

p0:
1 11 19 21 25 29 31 41
49 55 63 65 71 75 79 83
89 91 93 93 99

p1:
94 92 92 88 86 76 70 70
70 68 62 52 50 50 40 38
34 34 26 26 24 24 18 18
16 14 12 12 6

Concatenated List:
1 11 19 21 25 29 31 41
49 55 63 65 71 75 79 83
89 91 93 93 99 94 92 92
88 86 76 70 70 70 68 62
52 50 50 40 38 34 34 26
26 24 24 18 18 16 14 12
12 6

Circular List:
1 11 19 21 25 29 31 41
49 55 63 65 71 75 79 83
89 91 93 93 99 94 92 92
88 86 76 70 70 70 68 62
52 50 50 40 38 34 34 26
26 24 24 18 18 16 14 12
12 6

Eeny, meeny, miny, moe Result:
19 29 49 65 79 91 99 92
76 70 52 40 34 24 18 12
1 25 55 75 93 92 70 62
38 26 16 6 31 71 93 86
50 26 14 21 83 88 50 18
41 94 34 11 70 12 68 89
24 63

C:\Users\user\Desktop\新增資料夾\final_exam_linkedlist>
```

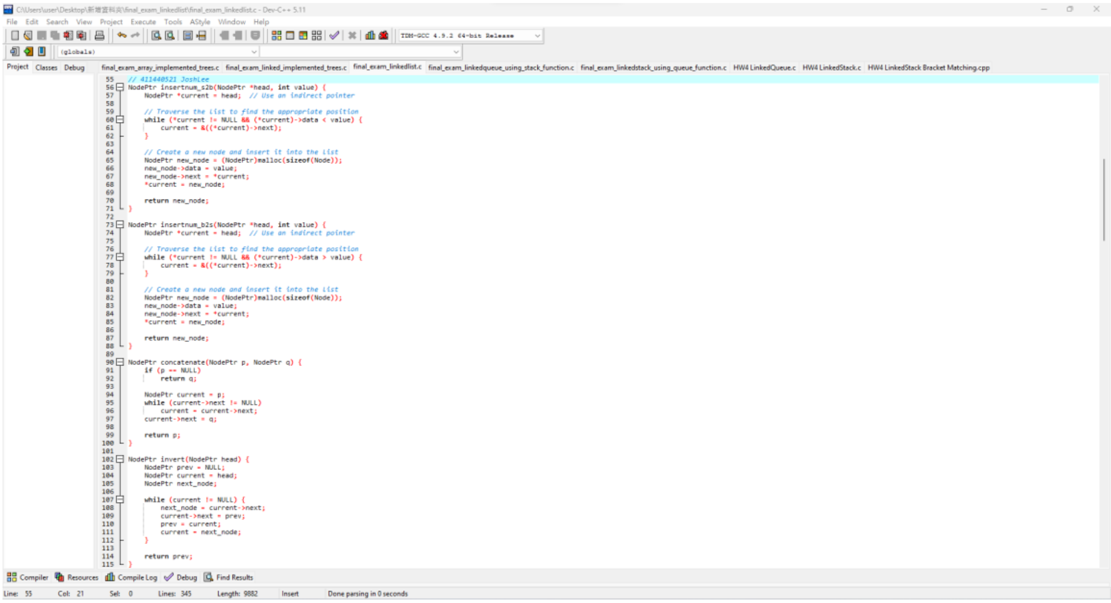

C:\Users\user\Desktop\新增資料夾\final_exam_linkedlist\final_exam_linkedlist.c - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_linkedlist.c
Compiler Resources Compile Log Debug Find Results
Line: 329 Col: 25 Sel: 0 Lines: 345 Length: 9938 Insert Done parsing in 0 seconds
C:\Windows\System32\cmd.e

```c
    // (sorted in ascending order -  from small to big),
    // and insert the even numbers into another created linear list (listPointer list2) one by one
    // (sorted in ascending order -  from big to small).
    // (Note: Be sure to use the two functions void insertnum_s2b(listPointer *p, int x) and void inser
    random_number_generator(&list, 100, 1, 50);
    printL(list);
    printf("\n");

    p0 = filter_and_insert(list, is_odd, insertnum_s2b);
    p1 = filter_and_insert(list, is_even, insertnum_b2s);

    // 2. output the linear list data from list1 and list2
    // (one by one from the first to the last, one space in between, and 10 numbers in one line).
    // (Note: Be sure to use the function void printL(listPointer p). )
    printf("p0:\n");
    printL(p0);
    printf("\n");
    printf("p1:\n");
    printL(p1);
    printf("\n");

    // 3.
    // produce a new linear list (listPointer p) that contains the linear list list1 followed by the li
    // and output data from the new linear list list (one by one from the first to the last, one space
    // (Note: Be sure to use the function listPointer concatenate(listPointer p1, listPointer p2). )
    NodePtr p = concatenate(p0, p1);
    printf("Concatenated List:\n");
    printL(p);
    printf("\n");

    // 4.
    // change the linear list invlist to a circular list.
    // (Note: Be sure to use the function listPointer Lin2Cir(listPointer p). )
    p = Lin2Cir(p);

    // 5.
    // output the circular list data
    // (one by one from the first to the last, one space in between, and 10 numbers in one line).
    // (Note: Be sure to use the function void printC(listPointer p). )
    printf("Circular List:\n");
    printC(p);
    printf("\n");
    // 411440521 JoshLee
    // 6.
    // After it becomes a circular list,
    // start counting from the first node.
    // When it reaches the sixth element,
    // use delete() to delete the node
    // and output the content of the node
    // until the entire circular list has no nodes.
    printf("Eeny, meeny, miny, moe Result:\n");
    NodePtr result = emmm(&p, 3);  // list was cleared
    printL(result);

    // Free memory
    free_list(&result);

    return 0;
}
```

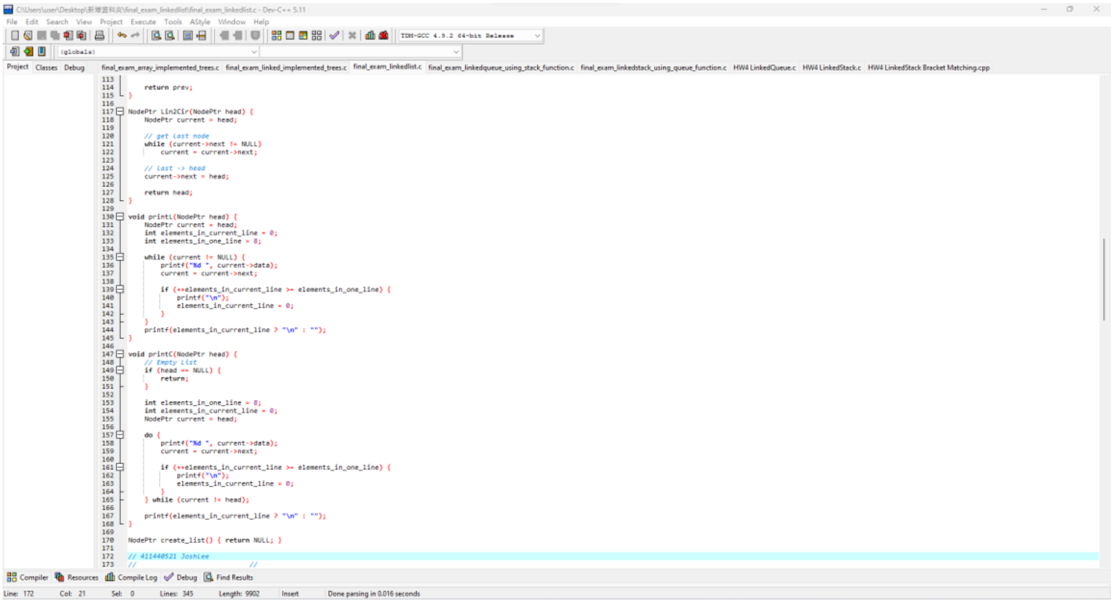
```
Eeny, meeny, miny, moe Result:
List is empty!
C:\Users\user\Desktop\新增資料夾\final_exam_linkedlist>final_exam_linkedlist.exe
1 6 11 12 12 14 16 18
18 19 21 24 24 25 26 26
29 31 34 34 38 40 41 49
50 50 52 55 62 63 65 68
70 70 70 71 75 76 79 83
86 88 89 91 92 92 93 93
94 99

p0:
1 11 19 21 25 29 31 41
49 55 63 65 71 75 79 83
89 91 93 93 99

p1:
94 92 92 88 86 76 70 70
70 68 62 52 50 50 40 38
34 34 26 26 24 24 18 18
16 14 12 12 6

Concatenated List:
1 11 19 21 25 29 31 41
49 55 63 65 71 75 79 83
89 91 93 93 99 94 92 92
88 86 76 70 70 70 68 62
52 50 50 40 38 34 34 26
26 24 24 18 18 16 14 12
12 6

Circular List:
1 11 19 21 25 29 31 41
49 55 63 65 71 75 79 83
89 91 93 93 99 94 92 92
88 86 76 70 70 70 68 62
52 50 50 40 38 34 34 26
26 24 24 18 18 16 14 12
12 6

Eeny, meeny, miny, moe Result:
19 29 49 65 79 91 99 92
76 70 52 40 34 24 18 12
1 25 55 75 93 92 70 62
38 26 16 6 31 71 93 86
50 26 14 21 83 88 50 18
41 94 34 11 70 12 68 89
24 63

C:\Users\user\Desktop\新增資料夾\final_exam_linkedlist>
```


C:\Users\user\Desktop\新增資料夾\final_exam_linkedlist\final_exam_linkedlist.c - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_linkedlist.c
Compiler Resources Compile Log Debug Find Results
Line: 12 Col: 21 Sel: 0 Lines: 345 Length: 9938 Insert Done parsing in 0 seconds

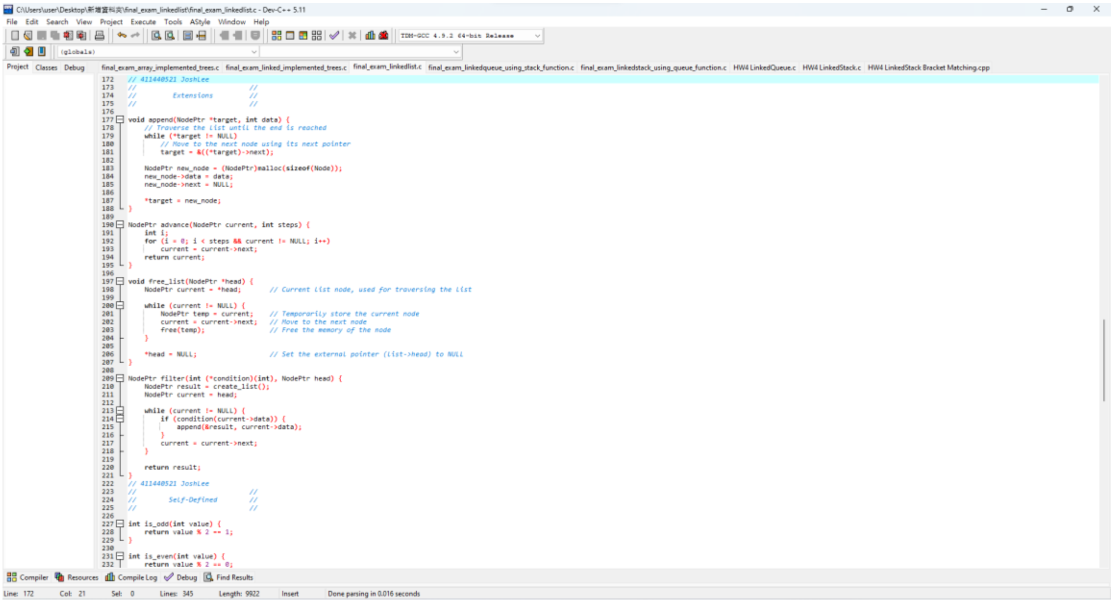
```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

// Define List Node
typedef struct _Node *NodePtr;
typedef struct _Node {
    int data;                   // Store value of the node
    NodePtr next;               // Store address of the next node
} Node;

// 411440521 JoshLee
void insert(NodePtr *target, int data) {  // double pointer for adding first element
    NodePtr new_node = (NodePtr)malloc(sizeof(Node));
    new_node->data = data;
    new_node->next = NULL;

    // for empty list
    if (*target == NULL) {
        *target = new_node;
        return;
    }

    // for non-empty list
    new_node->next = (*target)->next;
    (*target)->next = new_node;
}

int delete(NodePtr *target) {  // double pointer for deleting last element
    // for empty
    if (*target == NULL) {
        printf("List is empty!");
        exit(1);
    }

    // for the last element
    if ((*target)->next == NULL ||      // for normal list
        (*target)->next == *target) {  // for circular list
        NodePtr deleted = *target;
        int data = deleted->data;

        *target = NULL;
        free(deleted);
        return data;
    }

    // for other elements
    NodePtr deleted = (*target)->next;
    int data = deleted->data;

    (*target)->next = deleted->next;
    free(deleted);
    return data;
}
// 411440521 JoshLee
NodePtr insertnum_s2b(NodePtr *head, int value) {
    NodePtr *current = head;  // Use an indirect pointer

    // Traverse the list to find the appropriate position
    while (*current != NULL && (*current)->data < value) {
        current = &((*current)->next);
```


C:\Users\user\Desktop\新增資料夾\final_exam_linkedlist\final_exam_linkedlist.c - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_array_implemented_trees.c final_exam_linked_implemented_trees.c final_exam_linkedlist.c final_exam_linkedqueue_using_stack_function.c final_exam_linkedstack_using_queue_function.c HW4 LinkedQueue.c HW4 LinkedStack.c HW4 LinkedStack Bracket Matching.cpp
Compiler Resources Compile Log Debug Find Results
Line: 55 Col: 21 Sel: 0 Lines: 345 Length: 9882 Insert Done parsing in 0 seconds

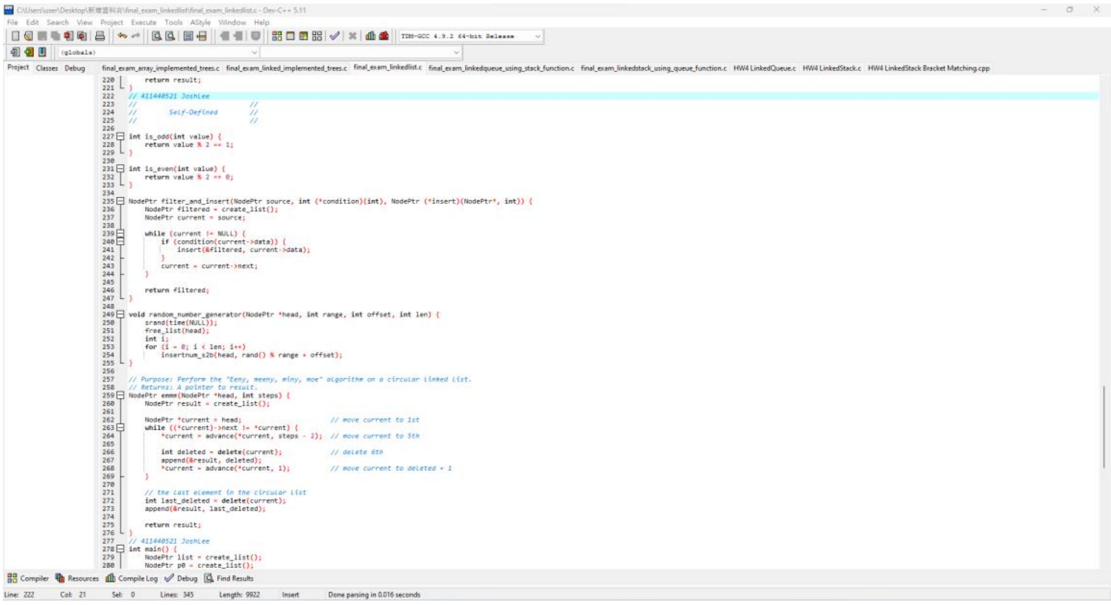
```c
// 411440521 JoshLee
NodePtr insertnum_s2b(NodePtr *head, int value) {
    NodePtr *current = head;  // Use an indirect pointer

    // Traverse the list to find the appropriate position
    while (*current != NULL && (*current)->data < value) {
        current = &((*current)->next);
    }

    // Create a new node and insert it into the list
    NodePtr new_node = (NodePtr)malloc(sizeof(Node));
    new_node->data = value;
    new_node->next = *current;
    *current = new_node;

    return new_node;
}

NodePtr insertnum_b2s(NodePtr *head, int value) {
    NodePtr *current = head;  // Use an indirect pointer

    // Traverse the list to find the appropriate position
    while (*current != NULL && (*current)->data > value) {
        current = &((*current)->next);
    }

    // Create a new node and insert it into the list
    NodePtr new_node = (NodePtr)malloc(sizeof(Node));
    new_node->data = value;
    new_node->next = *current;
    *current = new_node;

    return new_node;
}

NodePtr concatenate(NodePtr p, NodePtr q) {
    if (p == NULL)
        return q;

    NodePtr current = p;
    while (current->next != NULL)
        current = current->next;
    current->next = q;

    return p;
}

NodePtr invert(NodePtr head) {
    NodePtr prev = NULL;
    NodePtr current = head;
    NodePtr next_node;

    while (current != NULL) {
        next_node = current->next;
        current->next = prev;
        prev = current;
        current = next_node;
    }

    return prev;
}
```


C:\Users\user\Desktop\新增資料夾\final_exam_linkedlist\final_exam_linkedlist.c - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_array_implemented_trees.c final_exam_linked_implemented_trees.c final_exam_linkedlist.c final_exam_linkedqueue_using_stack_function.c final_exam_linkedstack_using_queue_function.c HW4 LinkedQueue.c HW4 LinkedStack.c HW4 LinkedStack Bracket Matching.cpp
Compiler Resources Compile Log Debug Find Results
Line: 172 Col: 21 Sel: 0 Lines: 345 Length: 9902 Insert Done parsing in 0.016 seconds

```c

    return prev;
}

NodePtr Lin2Cir(NodePtr head) {
    NodePtr current = head;

    // get last node
    while (current->next != NULL)
        current = current->next;

    // last -> head
    current->next = head;

    return head;
}

void printL(NodePtr head) {
    NodePtr current = head;
    int elements_in_current_line = 0;
    int elements_in_one_line = 8;

    while (current != NULL) {
        printf("%d ", current->data);
        current = current->next;

        if (++elements_in_current_line >= elements_in_one_line) {
            printf("\n");
            elements_in_current_line = 0;
        }
    }
    printf(elements_in_current_line ? "\n" : "");
}

void printC(NodePtr head) {
    // Empty List
    if (head == NULL) {
        return;
    }

    int elements_in_one_line = 8;
    int elements_in_current_line = 0;
    NodePtr current = head;

    do {
        printf("%d ", current->data);
        current = current->next;

        if (++elements_in_current_line >= elements_in_one_line) {
            printf("\n");
            elements_in_current_line = 0;
        }
    } while (current != head);

    printf(elements_in_current_line ? "\n" : "");
}

NodePtr create_list() { return NULL; }

// 411440521 JoshLee
//                              //
```

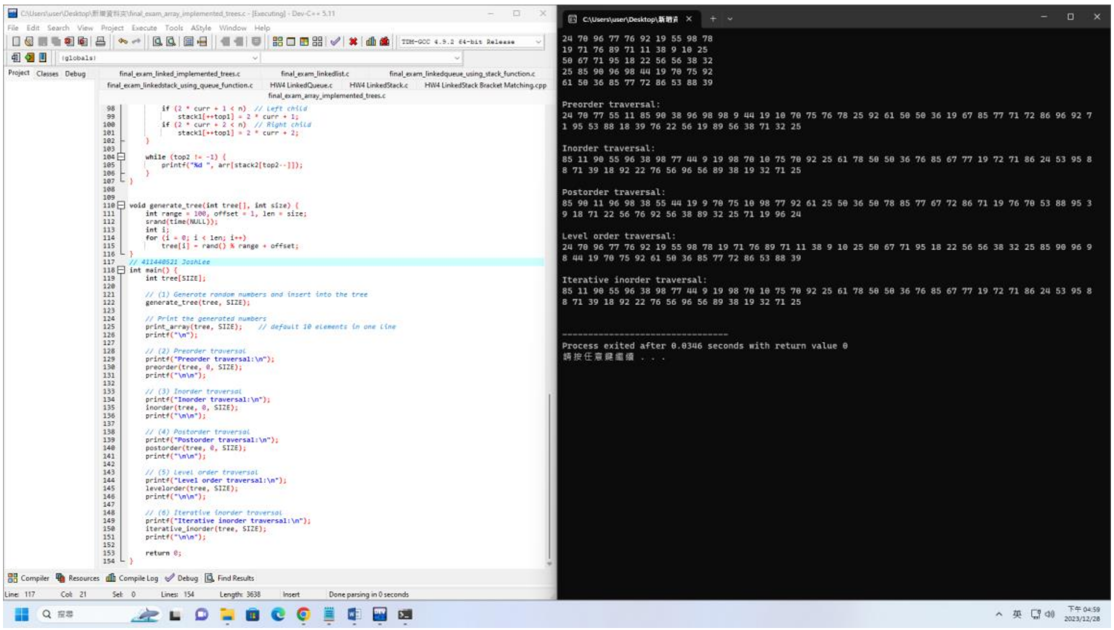

C:\Users\user\Desktop\新增資料夾\final_exam_linkedlist\final_exam_linkedlist.c - Dev-C++ 5.11
File
Edit
Search
View
Project
Execute
Tools
AStyle
Window
Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project
Classes
Debug
final_exam_array_implemented_trees.c
final_exam_linked_implemented_trees.c
final_exam_linkedlist.c
final_exam_linkedqueue_using_stack_function.c
final_exam_linkedstack_using_queue_function.c
HW4 LinkedQueue.c
HW4 LinkedStack.c
HW4 LinkedStack Bracket Matching.cpp
Compiler
Resources
Compile Log
Debug
Find Results
Line: 172
Col: 21
Sel: 0
Lines: 345
Length: 9922
Insert
Done parsing in 0.016 seconds

```c
// 411440521 JoshLee
//                          //
//         Extensions       //
//                          //

void append(NodePtr *target, int data) {
    // Traverse the list until the end is reached
    while (*target != NULL)
        // Move to the next node using its next pointer
        target = &((*target)->next);

    NodePtr new_node = (NodePtr)malloc(sizeof(Node));
    new_node->data = data;
    new_node->next = NULL;

    *target = new_node;
}

NodePtr advance(NodePtr current, int steps) {
    int i;
    for (i = 0; i < steps && current != NULL; i++)
        current = current->next;
    return current;
}

void free_list(NodePtr *head) {
    NodePtr current = *head;       // Current list node, used for traversing the list

    while (current != NULL) {
        NodePtr temp = current;    // Temporarily store the current node
        current = current->next;   // Move to the next node
        free(temp);                // Free the memory of the node
    }

    *head = NULL;                  // Set the external pointer (list->head) to NULL
}

NodePtr filter(int (*condition)(int), NodePtr head) {
    NodePtr result = create_list();
    NodePtr current = head;

    while (current != NULL) {
        if (condition(current->data)) {
            append(&result, current->data);
        }
        current = current->next;
    }

    return result;
}
// 411440521 JoshLee
//                          //
//        Self-Defined      //
//                          //

int is_odd(int value) {
    return value % 2 == 1;
}

int is_even(int value) {
    return value % 2 == 0;
```


C:\Users\user\Desktop\新增資料夾\final_exam_linkedlist\final_exam_linkedlist.c - Dev-C++ 5.11
File
Edit
Search
View
Project
Execute
Tools
AStyle
Window
Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project
Classes
Debug
final_exam_array_implemented_trees.c
final_exam_linked_implemented_trees.c
final_exam_linkedlist.c
final_exam_linkedqueue_using_stack_function.c
final_exam_linkedstack_using_queue_function.c
HW4 LinkedQueue.c
HW4 LinkedStack.c
HW4 LinkedStack Bracket Matching.cpp
Compiler
Resources
Compile Log
Debug
Find Results
Line: 222
Col: 21
Sel: 0
Lines: 345
Length: 9922
Insert
Done parsing in 0.016 seconds

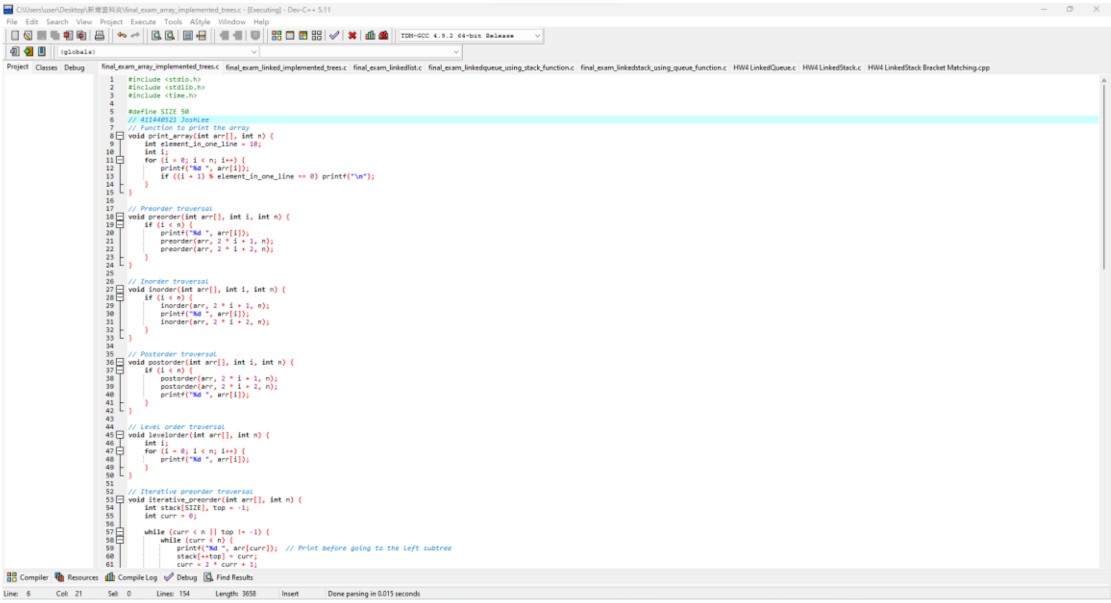
```c
    return result;
}
// 411440521 JoshLee
//                          //
//        Self-Defined      //
//                          //

int is_odd(int value) {
    return value % 2 == 1;
}

int is_even(int value) {
    return value % 2 == 0;
}

NodePtr filter_and_insert(NodePtr source, int (*condition)(int), NodePtr (*insert)(NodePtr*, int)) {
    NodePtr filtered = create_list();
    NodePtr current = source;

    while (current != NULL) {
        if (condition(current->data)) {
            insert(&filtered, current->data);
        }
        current = current->next;
    }

    return filtered;
}

void random_number_generator(NodePtr *head, int range, int offset, int len) {
    srand(time(NULL));
    free_list(head);
    int i;
    for (i = 0; i < len; i++)
        insertnum_s2b(head, rand() % range + offset);
}

// Purpose: Perform the "Eeny, meeny, miny, moe" algorithm on a circular linked list.
// Returns: A pointer to result.
NodePtr emmm(NodePtr *head, int steps) {
    NodePtr result = create_list();

    NodePtr *current = head;                        // move current to 1st
    while ((*current)->next != *current) {
        *current = advance(*current, steps - 2);   // move current to 5th

        int deleted = delete(current);              // delete 6th
        append(&result, deleted);
        *current = advance(*current, 1);            // move current to deleted + 1
    }

    // the last element in the circular list
    int last_deleted = delete(current);
    append(&result, last_deleted);

    return result;
}
// 411440521 JoshLee
int main() {
    NodePtr list = create_list();
    NodePtr p0 = create_list();
```

4. Tree

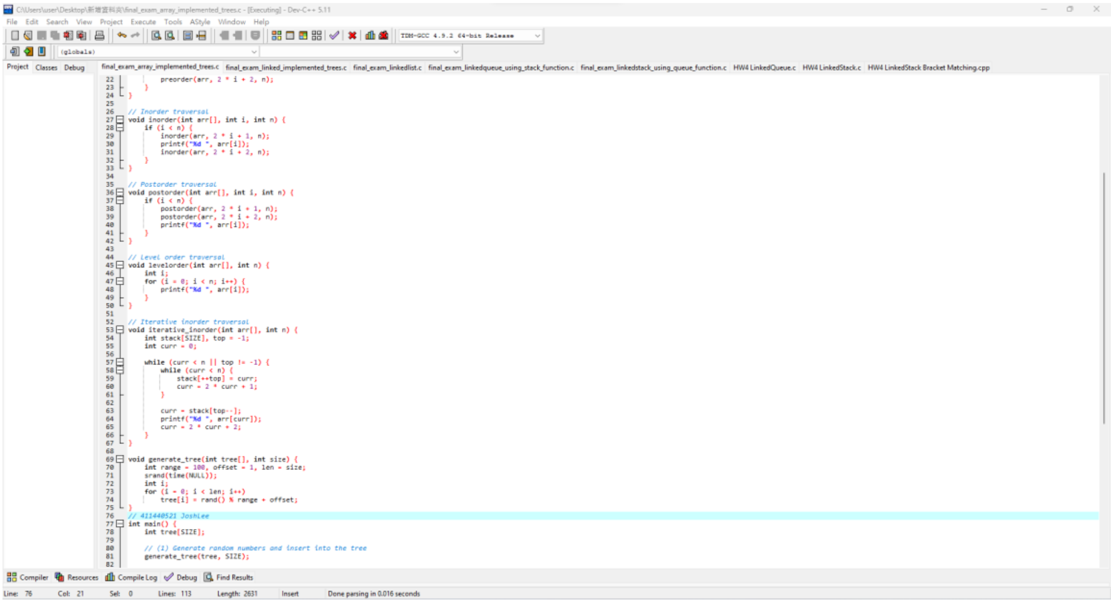

C:\Users\user\Desktop\新增資料夾\final_exam_array_implemented_trees.c - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_linked_implemented_trees.c
final_exam_linkedlist.c
final_exam_linkedqueue_using_stack_function.c
final_exam_linkedstack_using_queue_function.c
HW4 LinkedQueue.c
HW4 LinkedStack.c
HW4 LinkedStack Bracket Matching.cpp
final_exam_array_implemented_trees.c
Compiler Resources Compile Log Debug Find Results
Line: 117 Col: 21 Sel: 0 Lines: 154 Length: 3638 Insert Done parsing in 0 seconds
C:\Users\user\Desktop\新增資
下午 04:50
2023/12/28

```c
            if (2 * curr + 1 < n)  // Left child
                stack1[++top1] = 2 * curr + 1;
            if (2 * curr + 2 < n)  // Right child
                stack1[++top1] = 2 * curr + 2;
        }

    while (top2 != -1) {
        printf("%d ", arr[stack2[top2--]]);
    }
}


void generate_tree(int tree[], int size) {
    int range = 100, offset = 1, len = size;
    srand(time(NULL));
    int i;
    for (i = 0; i < len; i++)
        tree[i] = rand() % range + offset;
}
// 411440521 JoshLee
int main() {
    int tree[SIZE];

    // (1) Generate random numbers and insert into the tree
    generate_tree(tree, SIZE);

    // Print the generated numbers
    print_array(tree, SIZE);    // default 10 elements in one line
    printf("\n");

    // (2) Preorder traversal
    printf("Preorder traversal:\n");
    preorder(tree, 0, SIZE);
    printf("\n\n");

    // (3) Inorder traversal
    printf("Inorder traversal:\n");
    inorder(tree, 0, SIZE);
    printf("\n\n");

    // (4) Postorder traversal
    printf("Postorder traversal:\n");
    postorder(tree, 0, SIZE);
    printf("\n\n");

    // (5) Level order traversal
    printf("Level order traversal:\n");
    levelorder(tree, SIZE);
    printf("\n\n");

    // (6) Iterative inorder traversal
    printf("Iterative inorder traversal:\n");
    iterative_inorder(tree, SIZE);
    printf("\n\n");

    return 0;
}
```

```
24 70 96 77 76 92 19 55 98 78
19 71 76 89 71 11 38 9 10 25
50 67 71 95 18 22 56 56 38 32
25 85 90 96 98 44 19 70 75 92
61 50 36 85 77 72 86 53 88 39

Preorder traversal:
24 70 77 55 11 85 90 38 96 98 98 9 44 19 10 70 75 76 78 25 92 61 50 50 36 19 67 85 77 71 72 86 96 92 7
1 95 53 88 18 39 76 22 56 19 89 56 38 71 32 25

Inorder traversal:
85 11 90 55 96 38 98 77 44 9 19 98 70 10 75 70 92 25 61 78 50 50 36 76 85 67 77 19 72 71 86 24 53 95 8
8 71 39 18 92 22 76 56 96 56 89 38 19 32 71 25

Postorder traversal:
85 90 11 96 98 38 55 44 19 9 70 75 10 98 77 92 61 25 50 36 50 78 85 77 67 72 86 71 19 76 70 53 88 95 3
9 18 71 22 56 76 92 56 38 89 32 25 71 19 96 24

Level order traversal:
24 70 96 77 76 92 19 55 98 78 19 71 76 89 71 11 38 9 10 25 50 67 71 95 18 22 56 56 38 32 25 85 90 96 9
8 44 19 70 75 92 61 50 36 85 77 72 86 53 88 39

Iterative inorder traversal:
85 11 90 55 96 38 98 77 44 9 19 98 70 10 75 70 92 25 61 78 50 50 36 76 85 67 77 19 72 71 86 24 53 95 8
8 71 39 18 92 22 76 56 96 56 89 38 19 32 71 25


--------------------------------
Process exited after 0.0346 seconds with return value 0
請按任意鍵繼續 . . .
```


C:\Users\user\Desktop\新增資料夾\final_exam_array_implemented_trees.c - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
final_exam_array_implemented_trees.c
final_exam_linked_implemented_trees.c
final_exam_linkedlist.c
final_exam_linkedqueue_using_stack_function.c
final_exam_linkedstack_using_queue_function.c
HW4 LinkedQueue.c
HW4 LinkedStack.c
HW4 LinkedStack Bracket Matching.cpp
Compiler Resources Compile Log Debug Find Results
Line: 6 Col: 21 Sel: 0 Lines: 154 Length: 3658 Insert Done parsing in 0.015 seconds

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

#define SIZE 50
// 411440521 JoshLee
// Function to print the array
void print_array(int arr[], int n) {
    int element_in_one_line = 10;
    int i;
    for (i = 0; i < n; i++) {
        printf("%d ", arr[i]);
        if ((i + 1) % element_in_one_line == 0) printf("\n");
    }
}

// Preorder traversal
void preorder(int arr[], int i, int n) {
    if (i < n) {
        printf("%d ", arr[i]);
        preorder(arr, 2 * i + 1, n);
        preorder(arr, 2 * i + 2, n);
    }
}

// Inorder traversal
void inorder(int arr[], int i, int n) {
    if (i < n) {
        inorder(arr, 2 * i + 1, n);
        printf("%d ", arr[i]);
        inorder(arr, 2 * i + 2, n);
    }
}

// Postorder traversal
void postorder(int arr[], int i, int n) {
    if (i < n) {
        postorder(arr, 2 * i + 1, n);
        postorder(arr, 2 * i + 2, n);
        printf("%d ", arr[i]);
    }
}

// Level order traversal
void levelorder(int arr[], int n) {
    int i;
    for (i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
}

// Iterative preorder traversal
void iterative_preorder(int arr[], int n) {
    int stack[SIZE], top = -1;
    int curr = 0;

    while (curr < n || top != -1) {
        while (curr < n) {
            printf("%d ", arr[curr]);  // Print before going to the left subtree
            stack[++top] = curr;
            curr = 2 * curr + 1;
```


C:\Users\user\Desktop\新增資料夾\final_exam_array_implemented_trees.c - [Executing] - Dev-C++ 5.11
File
Edit
Search
View
Project
Execute
Tools
AStyle
Window
Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project
Classes
Debug
final_exam_array_implemented_trees.c
final_exam_linked_implemented_trees.c
final_exam_linkedlist.c
final_exam_linkedqueue_using_stack_function.c
final_exam_linkedstack_using_queue_function.c
HW4 LinkedQueue.c
HW4 LinkedStack.c
HW4 LinkedStack Bracket Matching.cpp
Compiler
Resources
Compile Log
Debug
Find Results

```c
        preorder(arr, 2 * i + 2, n);
    }
}

// Inorder traversal
void inorder(int arr[], int i, int n) {
    if (i < n) {
        inorder(arr, 2 * i + 1, n);
        printf("%d ", arr[i]);
        inorder(arr, 2 * i + 2, n);
    }
}

// Postorder traversal
void postorder(int arr[], int i, int n) {
    if (i < n) {
        postorder(arr, 2 * i + 1, n);
        postorder(arr, 2 * i + 2, n);
        printf("%d ", arr[i]);
    }
}

// Level order traversal
void levelorder(int arr[], int n) {
    int i;
    for (i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
}

// Iterative inorder traversal
void iterative_inorder(int arr[], int n) {
    int stack[SIZE], top = -1;
    int curr = 0;

    while (curr < n || top != -1) {
        while (curr < n) {
            stack[++top] = curr;
            curr = 2 * curr + 1;
        }

        curr = stack[top--];
        printf("%d ", arr[curr]);
        curr = 2 * curr + 2;
    }
}

void generate_tree(int tree[], int size) {
    int range = 100, offset = 1, len = size;
    srand(time(NULL));
    int i;
    for (i = 0; i < len; i++)
        tree[i] = rand() % range + offset;
}
// 411440521 JoshLee
int main() {
    int tree[SIZE];

    // (1) Generate random numbers and insert into the tree
    generate_tree(tree, SIZE);

```

Line: 76   Col: 21   Sel: 0   Lines: 113   Length: 2631   Insert   Done parsing in 0.016 seconds