COMP 2012H Midterm Exam - Fall 2019 - HKUST

1)

(a) F

(b) F

(c) F T

(d) T

(e) F?

(f) F

(g) T

(h) T?

(i) F

(j) T

2)

(a)

const Point& plus\_equal(Point& a, const Point& b) {

a.x += b.x;

a.y += b.y;

return a;

}

(b)

const Point& assign(Point& a, const Point& b) {

if (&a != &b) {

a.x = b.x;

a.y = b.y;

}

return a;

}

3)

int& a, int& b, int\*& c

4)

(a) 30

(b) Returns the largest between

- the maximum contiguous sum of the array, and

- the maximum contiguous sum of the negative of the array plus the sum of elements within the array

5)

line 18: causes runtime error on line 45, since ...

replace by

for (int i = 0; i < arr.size; i++) {

pArr->p[i] = arr.p[i];

}

line 46:

add

delete &copy;

6)

(a)

void mark(int v, int n, bool visit[], bool adj[][100]) {

visit[v] = true;

for (int i = 0; i < n; i++) {

if (!adj[v][i]) continue;

if (visit[i]) continue;

mark(i, n, visit, adj);

}

}

(b)

int unvisited = -1;

do {

unvisited = -1;

for (int i = 0; i < n; i++) {

if (!visit[i]) {

unvisited = i;

break;

}

}

if (visited == -1) return;

mark(unvisited, n, visit, adj);

cnt++;

} while (visited != -1);

CDLL create() {

CDLL cdll;

cdll.sentinel = new CDLL\_Node;

cdll.sentinel->prev = cdll.sentinel->next = cdll.sentinel;

return cdll;

}

void destroy(CDLL& cdll) {

CDLL\_Node\* node = cdll.sentinel->next;

while (node != cdll.sentinel) {

node = node->next;

delete node->prev;

}

delete cdll.sentinel;

}

void print\_forward(const CDLL& cdll) {

CDLL\_Node\* node = cdll.sentinel->next;

while (node != cdll.sentinel) {

cout << node->data << " ";

node = node->next;

}

}

void print\_reverse(const CDLL& cdll) {

CDLL\_Node\* node = cdll.sentinel->prev;

while (node != cdll.sentinel) {

cout << node->data << " ";

node = node->prev;

}

}

bool is\_empty(const CDLL& cdll) {

return cdll.sentinel->next == cdll.sentinel;

}

CDLL\_Node\* get\_node\_at\_index(const CDLL& cdll, unsigned int index) {

CDLL\_Node\* node = cdll.sentinel->next;

while (index > 0 && node != cdll.sentinel) {

node = node->next;

index--;

}

if (node == cdll.sentinel) {

return cdll.sentinel;

} else {

return node;

}

}

void insert(CDLL& cdll, unsigned int index, int data) {

CDLL\_Node\* node = cdll.sentinel;

if (index == 0) {

CDLL\_Node\* new\_node = new CDLL\_Node;

new\_node->data = data;

new\_node->prev = cdll.sentinel;

new\_node->next = cdll.sentinel->next;

cdll.sentinel->next->prev = new\_node;

cdll.sentinel->next = new\_node;

return;

}

while (index > 0 && node != cdll.sentinel) {

index--;

node = node->next;

}

if (cdll.sentinel == node) return;

CDLL\_Node\* new\_node = new CDLL\_Node;

new\_node->data = data;

new\_node->prev = node;

new\_node->next = node->next;

node->next->prev = new\_node;

node->next = new\_node;

}

...