DSA Homework 2

B03902089 林良翰

**2.1 (1)**

The variable ‘c’ will be deleted by default when the function ‘sub1’ is returned. As a result, there would be run time error after executing ‘sub1’ because the variable ‘c’ isn’t exist,

**2.1 (2)**

The problem is that we have to delete the variable ‘pc’, or it will cause memory leak after the whole program closed.

**2.2 (1)**

bool repeat(int i, int B[], int max, int pos, int times) {

if (times >= 2) return true;

else if (pos >= max) return false;

else if (i == B[pos]) {

return repeat(i, B, max, pos+1, times+1);

} else {

pos++;

return repeat(i, B, max, pos+1, times);

}

}

bool find(int B[]) {

int repeated = 0;

for (int i = 1 ; i <= n-5 ; i++) {

if ( repeat(i, B, n, 0, 0)) {

repeated++;

}

if ( repeated == 5) return true;

}

return false;

}

**2.2 (2)**

**Get from the matrix :**

int get(int array[], int row, int column) {

int n = 0;

for (int i = 1 ; i < row ; i++) n += i;

return array[n+column-1];

}

**Put to the matrix :**

void put(int array[], int row, int column, int input) {

int n = 0;

for (int i = 1 ; i < row ; i++) n += i;

array[n+column-1] = input;

}

**Memory layout :**

**2.2 (3)**

class LinkList {

public:

//data...

LinkList \*next;

};

bool isSame(LinkList \*L\_head, LinkList \*M\_head) {

LinkList \*start\_L = L\_head;

LinkList \*start\_M = M\_head;

int headsame = 0;

do {

if (start\_L->data == M\_head->data) {

headsame = 1;

break;

}

start\_L = start\_L->next;

} while (start\_L =! L\_head);

if (headsame) {

do {

if (start\_L->data != start\_M->data) return false;

start\_L = start\_L->next;

start\_M = start\_M->next;

} while (start\_L != L\_head);

return true;

} else return false;

}

**2.2 (4)**

void swap(int \*a, int \*b) {

int temp = \*a;

\*a = \*b;

\*b = temp;

}

void rearrange(int array[], int pos, int swapPos, int max) {

if (pos >= max || swapPos >= max) return;

else if (array[pos]%2 != 0) {

swap(&array[pos], &array[swapPos]);

rearrange(array, pos, swapPos+1, max);

} else

rearrange(array, pos+1, pos+2, max);

}

int main() {

int array[max];

//....process array

rearrange(array, 0, 1, max);

return 0;

}

**2.2 (5)**

int swapPos = 1;

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

if (array[i]%2 == 0) {

swapPos = i + 2;

continue;

} else if (array[i]%2 != 2) {

swap(array[i], array[swapPos]);

swapPos++;

}

}

2.3

2.4

(1) 讀檔：

我使用的是fscanf去讀取資料,而資料結構為”DATA” class為主體,當中有兩個class陣列分枝,分別是”DATA\_USER”和”DATA\_AD”,還有六個function以及兩個int,最後”DATA\_USER”和”DATA\_AD”當中包含了剩餘所有資料的vector結構。

DATA

DATA\_AD[ ]

DATA\_USER[ ]

Click

Impression

URL

Ad

Advertiser

Depth

Position

Query

Keyword

Title

Description

Click

Impression

User

Get(int&, int&, int&, int&, int&)

Clicked(int&)

Impressed(int&, int&)

Profit(int&, double&)

Quit()

Read()

max\_usr

max\_ad

由於許多function會要求以user為排列順序,或是常常查找user的資訊,因此我才以DATA\_USER[ ]為其中一個class分支,並以陣列的形式儲存,此時陣列的註標即為實際user的名稱。另外DATA\_AD[ ]是專為Profit這個function所設計的分支,因為他要求以Ad來做排序,而且他所需要的資料不多,因此當中只含了三個資料的vector。

另外，在讀檔過程中，會不斷更新max\_usr和max\_ad的資料，找出所有資料中最大的user和ad。

■ class

■ function

■ int

■ vector

(2) function實作：

(i) Get：直接找到DATA\_USER陣列當中的某項輸入的user資料，然後在把該user中所有的資料一一比對，最後把符合的資料的Click和Impression加總再輸出答案。

(ii) Clicked：直接找到DATA\_USER陣列當中的某項輸入的user資料，此時再宣告一個struct陣列，當中包含了Ad和Query，然後再做比對，把符合條件的資料全部放到struct裡面。比對結束之後，再把struct做qsort：先以Ad來sort，Ad相同則再以Query來sort，最後再依照剛剛排好的struct順序來輸出答案。

(iii) Impressed：使用巢狀迴圈來找出兩個user所含有共同的Ad，此時開一個暫存的struct，當中包含Ad、url、adv、key、title、des，然後再開一個vector種類的struct。每當有符合得資料時，則先把資料存到暫存struct，然後再push\_back到vector種類的struct。所有比對結束之後，再進行qsort，sort的順序分別為Ad、url、adv、key、title、des，結著依照剛剛排好的vector種類的struct進行輸出。

(iv)Profit：先開一個struct陣列，專門儲存每個user的Click和Ad的總和，接著直接找到DATA\_AD當中所輸入的Ad資料，開始將符合該Ad各個user的Click和Impression加總，最後再依照暫存的struct順序把符合Click/Impression要求的user輸出。

(3) 心得：

這次作業30%是經過某些大神的指導，70%是我自己在自幹，大神指導的東西包括vector的使用、資料結構的設計，而我自己則自幹function的實作、qsort的使用方法。由於我開始寫的時間比較早(應該算是很早)，可以說沒有甚麼團隊作戰的精神，到最後大家都跑來問我QQ，然後我只好把我獨自奮鬥的研究成果一個一個傳給(口述)大家…。儘管這次作業花了我許多時間，但是我真的學習到許多C++和C的工具使用方法，而且我還大量使用了#ifdef和#endif來debug，幫助我更有效的寫完作業，然後希望下次作業能夠簡單一些。