Федеральное агентство связи

Сибирский Государственный Университет Телекоммуникаций и

Информатики

СибГУТИ

Кафедра ПМиК

Лабораторная работа №12.1

«Слияние серий»

Выполнил: студент 1 курса группы ИП-013

Иванов.Л.Д

Проверил: старший преподаватель кафедры ПМиК

Солодов П.С.

Новосибирск

2021

Лабораторная работа № 12-1

1) Разработать процедуру слияния серий и протестировать на различных примерах (количество элементов в сливаемых списках должно быть не менее 10).

2) Для проверки правильности слияния вывести на экран количество серий и контрольную сумму для всех списков.

3) Предусмотреть подсчет фактического количества сравнений и перемещений (из стека в очередь), сравнить с теоретическими оценками.

**код:**

#include <cmath>

#include <ctime>

#include <iostream>

using namespace std;

struct stack {

stack \*next;

int data;

};

struct queue {

stack \*head{};

stack \*tail{};

};

void PrintList(stack \*head);

void FillInc(stack\*\* head, int n);

void FillDec(stack \*\*head, int n);

void FillRand(stack \*\*head, int n);

int CheckSum(stack \*head);

int RunNumber(stack \*head);

void Merge(stack \*a, int q, stack \*b, int r, stack \*\*c\_tail);

int C = 0, M = 0;

int main() {

stack \*head\_a, \*head\_b, \*head;

queue head\_c;

head\_a = NULL, head\_b = NULL;

int q, r;

cout << "q=: ";

cin >> q;

FillInc(&head\_a, q);

PrintList(head\_a);

cout << "Sum = " << CheckSum(head\_a) << endl;

cout << "series = " << RunNumber(head\_a) << endl;

cout << "r=: ";

cin >> r;

FillInc(&head\_b, r);

PrintList(head\_b);

cout << "Sum = " << CheckSum(head\_b) << endl;

cout << "series = " << RunNumber(head\_b) << endl;

head\_c.tail = (stack \*) &head\_c.head;

cout << "Merge" << endl;

Merge(head\_a, q, head\_b, r, &head\_c.tail);

PrintList(head\_c.head);

cout << "Sum = " << CheckSum(head\_c.head) << endl;

cout << "series = " << RunNumber(head\_c.head) << endl;

C = 0, M = 0;

return 0;

}

void PrintList(stack \*head) {

stack \*p = head;

while (p) {

cout << p->data << " ";

p = p->next;

}

cout << endl;

}

void FillInc(stack\*\* head, int n) {

stack\* p;

for (int i = n - 1; i > -1; --i) {

p = new stack;

p->data = i;

p->next = \*head;

\*head = p;

}

}

void FillDec(stack \*\*head, int n) {

stack \*p;

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

p = new stack;

p->data = i;

p->next = \*head;

\*head = p;

}

}

void FillRand(stack \*\*head, int n) {

stack \*p;

srand(time(nullptr));

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

p = new stack;

p->data = rand() % (2 \* n);

p->next = \*head;

\*head = p;

}

}

int CheckSum(stack \*head) {

int sum = 0;

stack \*p = head;

while (p) {

sum += p->data;

p = p->next;

}

return sum;

}

int RunNumber(stack \*head) {

int i = 1;

stack \*p = head;

while (p->next) {

if (p->data > p->next->data) {

i+=1;

}

p = p->next;

}

return i;

}

void Merge(stack \*a, int q, stack \*b, int r, stack \*\*c\_tail) {

stack \*p\_tail = \*c\_tail;

while (q != 0 && r != 0) {

++C;

if (a->data <= b->data) {

p\_tail->next = a;

p\_tail = a;

a = a->next;

++M;

--q;

} else {

p\_tail->next = b;

p\_tail = b;

b = b->next;

++M;

--r;

}

}

while (q > 0) {

p\_tail->next = a;

p\_tail = a;

a = a->next;

++M;

--q;

}

while (r > 0) {

++M;

p\_tail->next = b;

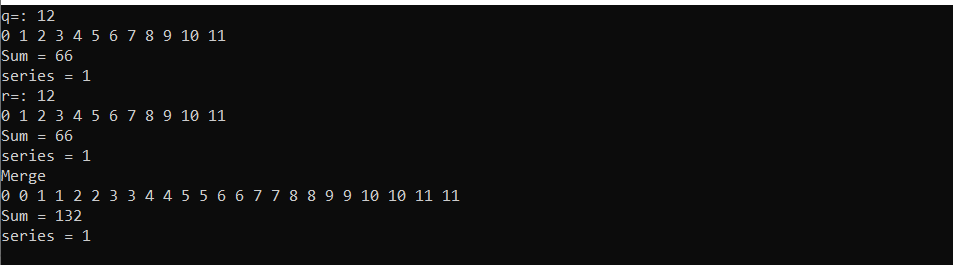
p\_tail = b;

b = b->next;

--r;

}

}



Вывод: Сегодня я изучил метод слияния списков, посчитал трудоемкость и сравнил её с теоритическими значениями.