

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

|  |  |
| --- | --- |
| Факультет | Информатика и вычислительная техника |
|  | (наименование факультета) |
| Кафедра | Кибербезопасность информационных систем |
|  | (наименование кафедры) |

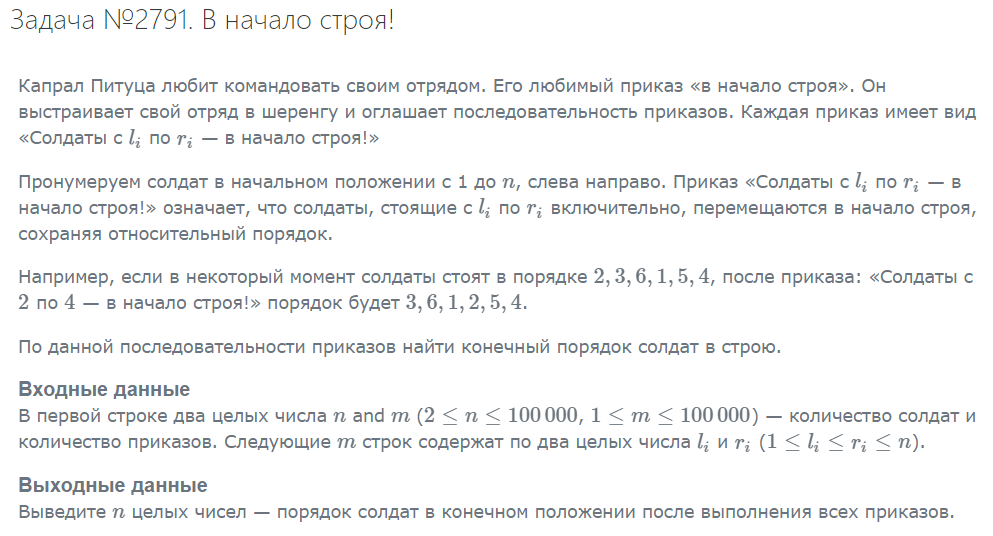
**ОТЧЕТ по лабораторным работам**

**по дисциплине “Методы программирования”**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Автор |  | | |  | Д.С. Минаков | | | |
|  | (подпись, дата) | | |  |  | | | |
| Обозначение | 10.05.01.230000.000 О | | Группа | | | | ВКБ33 |
| Направление подготовки | | 10.05.01 Компьютерная безопасность | | | | | |
| Профиль | Компьютерная безопасность | | | | | | |
| Преподаватель |  | | |  | | В.А. Савельев | |
|  | (подпись, дата) | | |  | |  | |

г. Ростов-на-Дону

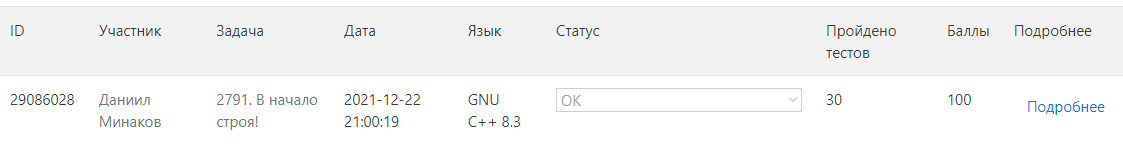
2021 год



Код:

|  |
| --- |
| #include <iostream> |
|  | #include <cstdlib> |
|  | using namespace std; |
|  |  |
|  | bool flag; |
|  |  |
|  | struct Item |
|  | { |
|  | int cnt, Value, Priority, Summa; |
|  | Item \*l, \*r; |
|  | Item() { } |
|  | Item(int Priority, int Value) : cnt(0), Value(Value), Priority(Priority), Summa(0), l(), r() { } |
|  | }; |
|  |  |
|  | typedef Item\* Pitem; |
|  | Pitem Tree, Ta, Tb, Tc; |
|  |  |
|  | void PrintTree(Pitem t) |
|  | { |
|  | if (!t) return; |
|  | PrintTree(t->l); |
|  | if (flag) printf(" "); |
|  | printf("%d", t->Value); |
|  | flag = 1; |
|  | PrintTree(t->r); |
|  | } |
|  |  |
|  | int cnt(Pitem t) |
|  | { |
|  | return t ? t->cnt : 0; |
|  | } |
|  |  |
|  | int GetSum(Pitem t) |
|  | { |
|  | if (t) return t->Summa; |
|  | return 0; |
|  | } |
|  |  |
|  | void update(Pitem t) |
|  | { |
|  | if (t) |
|  | { |
|  | t->cnt = 1 + cnt(t->l) + cnt(t->r); |
|  | t->Summa = t->Value + GetSum(t->l) + GetSum(t->r); |
|  | } |
|  | } |
|  |  |
|  | void Merge(Pitem l, Pitem r, Pitem &t) |
|  | { |
|  | if (!l || !r) t = l ? l : r; |
|  | else if (l->Priority > r->Priority) Merge(l->r, r, l->r), t = l; |
|  | else Merge(l, r->l, r->l), t = r; |
|  | update(t); |
|  | } |
|  |  |
|  | void Split(Pitem t, Pitem &l, Pitem &r, int pos) |
|  | { |
|  | if (!t) return void(l = r = 0); |
|  | if (pos <= cnt(t->l)) Split(t->l, l, t->l, pos), r = t; |
|  | else Split(t->r, t->r, r, pos - 1 - cnt(t->l)), l = t; |
|  | update(t); |
|  | } |
|  |  |
|  | //На позицию pos декартового дерева t вставляем вершину it. |
|  | void Insert(Pitem &t, Pitem it, int pos) |
|  | { |
|  | Pitem t1, t2; |
|  | Split(t, t1, t2, pos); |
|  | Merge(t1, it, t1); |
|  | Merge(t1, t2, t); |
|  | } |
|  |  |
|  | int main() { |
|  | int n, m; |
|  | scanf("%d %d", &n, &m); |
|  | for (int i = 0; i < n; i++) Insert(Tree, new Item(rand(), i + 1), i); |
|  |  |
|  | for (int i = 0; i < m; i++) |
|  | { |
|  | int l, r; |
|  | scanf("%d %d", &l, &r); |
|  | Split(Tree, Tb, Tc, r); |
|  | Split(Tb, Ta, Tb, l - 1); |
|  | Merge(Tb, Ta, Tree); |
|  | Merge(Tree, Tc, Tree); |
|  | } |
|  |  |
|  | PrintTree(Tree); |
|  | printf("\n"); |
|  | system("pause"); |
|  | return 0; |
|  | } |

Проверка:



Результат:

