```
//选取最好的战舰 关键代码
#define MAXDATAN 100
#define MAXN 100
#define INVALID_KEY '@'
string val_name[MAXDATAN];
int cur_val = 0;
int ship_n;
int val_n;
struct prop
{
    double val[MAXDATAN];
    double sum;
};
enum State
{
    mid = 0,
    end,
};
class DT
{
public:
    DT* children[MAXN];
    prop* data;
    char key;
    State state ;
    string name;
    DT() {
        for (int i = 0; i < MAXN; i++)
            children[i] = NULL;
        data = new prop;
        key = INVALID_KEY;
        state = mid;
        name = "";
        data->sum = 0;
    }
};
struct ElemType
{
    DT* data;
    double key;
};
struct Array
{
    int size;
    ElemType ships[MAXN];
};
void build(DT* T)
{
    cin >> val_name[cur_val];
```

```
    double val_weight;
    cin >> val_weight;
    for (int k = 0; k < ship_n; k++) {
        DT* p = T;
        string name;
        cin >> name;
        for (int i = 0; i < name.length(); i++) {
            int j = 0;
            while (p->children[j] != NULL && p->children[j]->key != name[i])
                j++;
            if (p->children[j] == NULL)
                p->children[j] = new DT;
            p = p->children[j];
            p->key = name[i];
            if (i == name.length()-1) {
                p->state = State::end;
                cin >> p->data->val[cur_val];
                p->data->val[cur_val] *= val_weight;
                p->data->sum += p->data->val[cur_val];
                p->name = name;
            }
        }
    }
    cur_val++;
    return;
}
void add(DT* T)
{
    cin >> val_name[cur_val];
    double val_weight;
    cin >> val_weight;
    for (int k = 0; k < ship_n; k++) {
        DT* p = T;
        string name;
        cin >> name;
        for (int i = 0; i < name.length(); i++) {
            int j = 0;
            while (p->children[j] != NULL && p->children[j]->key != name[i])
                j++;
            p = p->children[j];
            if (i == name.length()-1) {
                p->state = State::end;
                cin >> p->data->val[cur_val];
                p->data->val[cur_val] *= val_weight;
                p->data->sum += p->data->val[cur_val];
                p->name = name;
            }
        }
    }
```

```
    cur_val++;
    return;
}
void traverse(DT* T, Array& ships)
{
    if (T->state == State::end) {
        ElemType new_data = { T,T->data->sum };
        ships.ships[ships.size++] = new_data;
    }
    int i = 0;
    while (T->children[i] != NULL) {
        traverse(T->children[i], ships);
        i++;
    }
    return;
}
Array traverse_tree(DT T)
{
    Array ships;
    ships.size = 0;
    int i = 0;
    while (T.children[i] != NULL) {
        traverse(T.children[i], ships);
        i++;
    }
    return ships;
}
```