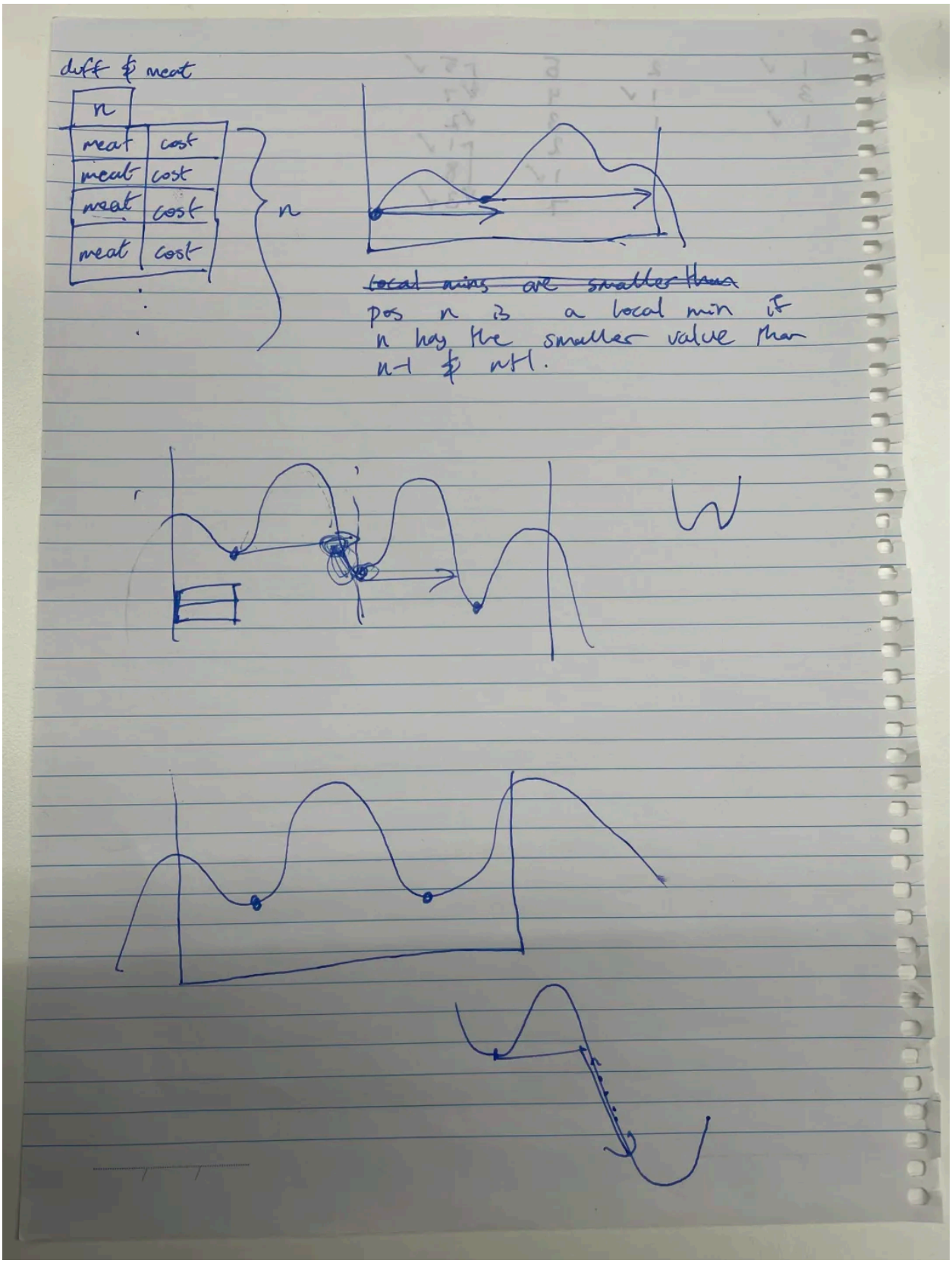
Problem 1:


diff $ meat
n
meat cost
meat cost
meat cost
meat cost
n
pos n is a local min if
n has the smaller value than
n-1 & n+1.

This problem is talking about how we would like to satisfy Duffy by keeping them full based on how much food they want denoted by a[i], however the price also fluctuates based on p[i]

The thing about this problem is that we're able to pre-purchase food so that when the prices are good we can just mass haul this price.

So my approach was looping through all of the prices and look for all "local minimums" based on this condition. if p[i -1] ≤ p[i] ≤ p[i +1] then p[i] is considered a local minimum which is a potential price that we would like a pre-purchase the meat for for upcoming days.

And then with these prices saved we just need to check if this indeed is better than any other upcoming p[k], and append this to the final cost.

The implementation looks like:

```
#include <iostream>
#include <algorithm>
#include <vector>
using namespace std;

int main() {

    int n;
    int total_cost = 0;
    int current_purchase_price;
    bool fixed_price = false;
    vector<int> a(n);
    vector<int> p(n);
    vector<bool> local_mins(n, false);

    cin >> n;
```

```
    for (int i = 0; i < n; i++) {
        cin >> a[i] >> p[i];
    }

    for (int i = 0; i < n; i++) {
        if (i == 0 && n > 1 && p[i] <= p[i + 1]) {
            local_mins[i] = true;

        } else if (p[i] <= p[i - 1] && p[i] <= p[i + 1]) {
            local_mins[i] = true;
        }
    }

    for (int i = 0; i < n; i++) {
        // get the new fixed prices
        if (local_mins[i] == true) {
            current_purchase_price = min(current_purchase_price, p[i]);
            fixed_price = true;
        }

        if (fixed_price == true) {
            total_cost += a[i] * min(current_purchase_price, p[i]);
        } else {
            total_cost += a[i] * p[i];
        }

    }

    cout << total_cost << endl;
}
```

retrospectively, I can see that this whole implementation could have theoretically been done in 1 for loop, and just buy it for the cheapest price that i have seen up until now. I hadn't thought about this because of the way the question was

phrased as it kind of made me think that i should pre-purchase food, for upcoming days

But technically in terms of the solution that is the same thing as buying the current days' quota for food with the cheapest price that I have seen previously.