```cpp
#include <vector>
#include <iostream>
#include <memory>

using std::shared_ptr;
using std::cout;
using std::endl;

using std::vector;

class Exceptions : public std::exception
{
    public:
        class BadInput{};
};

template <class T>
std::vector<T> slice(std::vector<T> vec, int start, int step ,int stop)
{
    if(start < 0 || stop < 0 || stop > vec.size() || step <= 0)
    {
        throw Exceptions::BadInput();
    }

    vector<T> sliced;
    if(start>=stop)
    {
        return  sliced;
    }

    for (int i = start ; i < stop ; i += step)
    {
        sliced.push_back(vec[i]);
    }

    return sliced;
}


class A
{
private:
public:
    A()=default;
    ~A()=default;
    vector<shared_ptr<int>> values;
    void add(int x)
    {
```

```
        shared_ptr<int> ptr(new int(x * 800));
        values.push_back(ptr);
    }
};


int main() {
    A a, sliced;
    a.add(0); a.add(1); a.add(2); a.add(3); a.add(4); a.add(5);
    sliced.values = slice(a.values, 1, 1, 4);
    *(sliced.values[0]) = 800;
    std::cout << *(a.values[1]) << std::endl;
    return 0;
}
```