```
#include <string>

string napis1 = "Zupa";
       2 = "pomidorowa";
cout << napis1 + napis2;
napis2.insert(napis2.begin(), ' ');
```

>>

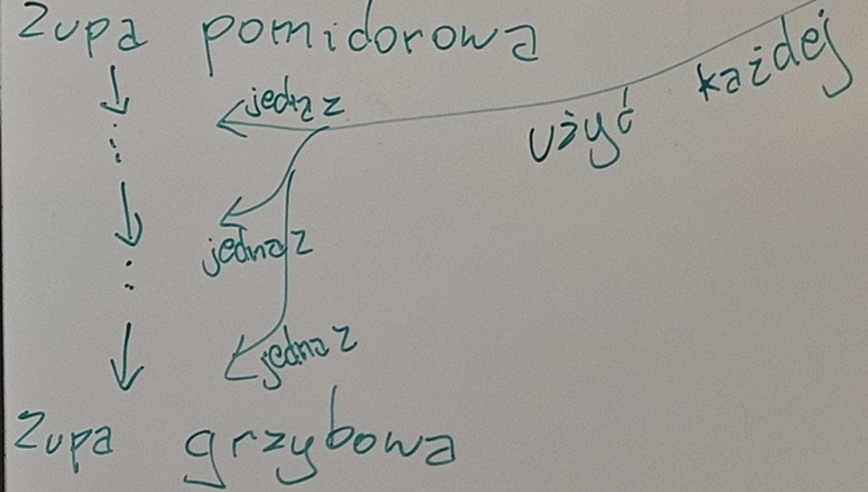
```
c:\user\franek> cmd
                Untitled2.cpp
```

To ...

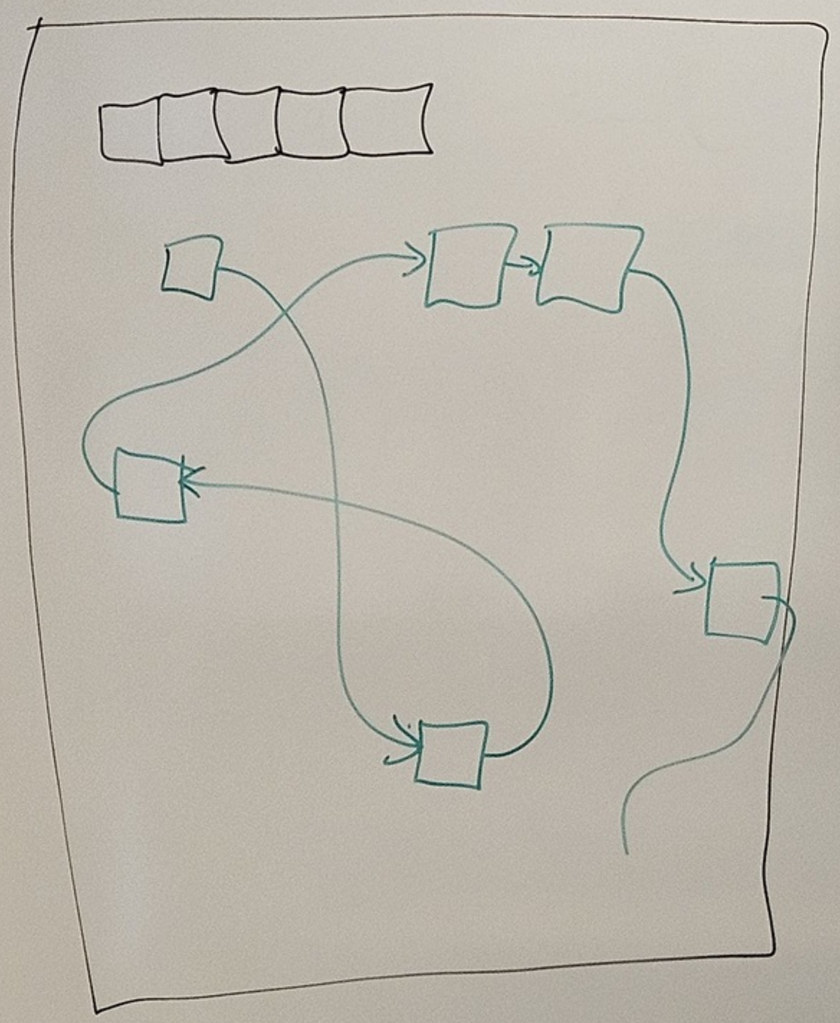
```
napis.erase(2,4)
     .replace(2,4,"dudu")
napis.substr(2,5)
napis == "Zupa"
napis.compare("Zupa")
```

2C

```
to_string(42)
stoi("42")
```




```
auto pos = napis2.find("owa")
if (pos != string::npos){
    cout << pos;
}
```



<T>

```
#include <list>

list <int>  lista = {3, 1, 4, 1, 5};

wyswietl_liste (lista, pre="\n");

int                    .end()
auto  pos = lista.begin();
cout << pos;
advance (pos, 2);
cout << pos;
lista.insert (pos, 303);
wyswietl_liste (lista);

lista.erase (pos)
     .pop_back()
     .front()

lista.push_back(-11)
     .push_front(-12)
     .remove(1)
lista1.merge(lista2)
```

wektor tab = {2, 1, 7, 8, 2, 8};

set <int> unikalne (tab.begin(), tab.end());

.size()
.erase(2)
.clear()

unikalne.insert(tab.begin(), tab.end());

w domu:

funkcja wyswietl_zbiór

```
#include <utility>
```

```
pair<int, string>    sprzedaz = {169, "Drapala"};
```

```
sprzedaz.first
        .second
```

```
dane[3].first
```

1. wektor z danymi o sprzedaży par

```
w.push_back( {283, "Skrzyniarz"} )
```

2. zsumować kwoty sprzedaży od wszystkich sprzedawców

```
struct pacjent {
    string imie;
    string nazw;
    int    wiek;
    int cis_dol;
    int cis_gor;
    int puls;
};
```

```
pacjent p;
```

SOR ⟹

```
[
  {"Aneta", "S", 18, 80, 120, 65},
  {"Jarek", "D", 19, 62, 139, 88},
]
```

#include <algorithm>

sort(tab.begin(), tab.end())
.rbegin(), .rend()

większe

bool wieksze(int a, int b) {
    return a > b;
}

#include <cmath>

pow(x, 2) sqrt(x) log(x) log10(x)

round(3.2)
3.7
3.5
-3.5
floor(3.9)
ceil(3.1)

<bits/stdc++.h>

