Gorączka zakupów

```
#void  #long_long
#standard_input
#standard_output
```

Czas na świąteczne zakupy i związane z nimi liczne promocje! W tym podrozdziale zajmiemy się nietrudnym zadaniem z konkursu **PIWO 2018/19**. Treść zadania znajdziecie tutaj.

Halinka wielokrotnie odwiedza ulubiony sklep i dokonuje w nim zakupów. Za każdym razem warunki zakupów są inne: dane są 4 parametry s, a, b, c (dodatnie liczby całkowite), które to określają. Wartość s oznacza kwotę pieniędzy, którymi dysponuje dziewczyna, zaś c to jednostkowa cena towaru. Promocja polega na tym, że za każde a sztuk kupionej odzieży Halinka otrzymuje gratis b sztuk tejże odzieży. Pytanie brzmi, z iloma sztukami w garści opuszcza sklep po kolejnych zakupach?

Program zorganizujemy tak, aby rozliczenie pojedynczej wizyty w sklepie odbywało się przy użyciu funkcji `shopping()` (wymawiaj: *szoping*). Funkcja ta nie zwraca żadnej wartości, więc w jej definicji pojawi się nowe słowo: `void` (wymawiaj: *wojd*):

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

void shopping()
{
  . . .
}

int main()
{
  int Z;
  cin >> Z;
  while(Z > 0)
  {
    shopping();
    Z--;
  }
}
```

Zmienna Z to ilość wizyt w sklepie i zarazem ilość obiegów pętli `while` w funkcji `main()`.

W funkcji `shopping()` musimy najpierw wczytać potrzebne dane i obliczyć, ile sztuk odzieży Halinka może „po Bożemu" kupić za swoją kasę (zmienna n):

```
int s, a, b, c;
cin >> s >> a >> b >> c;
int n = s / c;
```

Pamiętamy oczywiście, że takie dzielenie oznacza obcięcie części ułamkowej.

Skoro kupione jest n sztuk odzieży, to możemy z tego uformować n/a „zestawów promocyjnych", a za każdy zestaw Halinka dostanie dodatkowo b sztuk. Zatem ilość wszystkich gratisowych sztuk wynosi $(n/a) * b$ i funkcja `zakupy()` może wyglądać tak:

```
void shopping()
{
  int s, a, b, c;
  cin >> s >> a >> b >> c;
  int n = s / c;
  cout << n + (n / a) * b << endl;
}
```

Wygląda to nieźle, dopóki nie przyjrzymy się uważnie zakresowi danych wejściowych: każda z wczytanych liczb może osiągnąć wartość miliard, a najbardziej pesymistyczny przypadek będzie wtedy, gdy $s = b = 10^9$ oraz $a = c = 1$: mamy wtedy $n = n/a = 10^9$ i tę wartość musimy pomnożyć przez $b = 10^9$, co da nam 10^{18}, czyli liczbę o wiele za dużą dla zakresu typu `int`. Trzeba więc użyć typu całkowitego o większym zakresie: `long long`,* który radzi sobie z takimi liczbami. Ostatecznie cały program powinien wyglądać tak:

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

void shopping()
{
  long long s, a, b, c;
  cin >> s >> a >> b >> c;
  long long n = s / c;
  cout << n + (n / a) * b << endl;
}

int main()
{
  int Z;
  cin >> Z;
  while(Z > 0)
  {
    shopping();
    Z--;
  }
}
```

*W tym typie danych na jedną wartość przypada 8 bajtów, podczas gdy w typie `int` tylko 4.

Po uruchomieniu programu i wpisaniu przykładowych danych:

```
3
12 3 2 1
40 4 2 2
10000 23 4 3
```

otrzymujemy wynik:

```
20
30
3909
```

W rzeczywistości jednak obraz na ekranie jest inny. Ponieważ wpisywane dane przetwarzane są na bieżąco, więc po wpisaniu każdej linijki z czterema liczbami odpowiedni wynik jest od razu wypisywany na ekranie:

```
3
12 3 2 1
20
40 4 2 2
30
10000 23 4 3
3909
```

Jest to objaw jak najbardziej prawidłowy: po prostu na tym samym ekranie wyświetlane są dane wejściowe (strumień **standard input**), jak i wynik programu (strumień **standard output**, wymawiaj: *standard ałtput*) i ich treści przeplatają się. Jeśli nasz program jest uruchamiany na serwerze testerki, wtedy te dwa strumienie są odseparowane od siebie – dba o to odpowiednie oprogramowanie serwera.