Sample I/O

SCT Officers

October 19, 2020

The following are some I/O examples in various languages to get you started.

All programs read in a number N from the first line, proceed to read in N more numbers from the second line, and print their sum.

Sample input:

```
4
1 3 5 8
```

Sample output:

```
17
```

1 Java

Here I use a `BufferedReader` instead of a `Scanner`. It's much faster and much more reliable in the contest environment.

1.1 Standard I/O

```java
import java.util.*;
import java.io.*;

class sum {
    public static void main(String[] args) throws IOException {
        BufferedReader f = new BufferedReader(new InputStreamReader(System.in));
        int N = Integer.parseInt(f.readLine()); // read whole line
        int[] num = new int[N];
        StringTokenizer st = new StringTokenizer(f.readLine()); // split line by white space
        for(int k = 0; k < N; ++k) {
            num[k] = Integer.parseInt(st.nextToken());
        }
        int sum = 0;
        for(int k = 0; k < N; ++k) {
            sum += num[k];
        }
        System.out.println(sum);
        System.exit(0);
    }
}
```

1.2 File I/O

```java
import java.util.*;
import java.io.*;

class sum {
    public static void main(String[] args) throws IOException {
        BufferedReader f = new BufferedReader(new FileReader("sum.in"));
        PrintWriter out = new PrintWriter(new BufferedWriter(new FileWriter("sum.out")));
        int N = Integer.parseInt(f.readLine()); // read whole line
        int[] num = new int[N];
        StringTokenizer st = new StringTokenizer(f.readLine()); // split line by white space
        for(int k = 0; k < N; ++k) {
            num[k] = Integer.parseInt(st.nextToken());
        }
        int sum = 0;
        for(int k = 0; k < N; ++k) {
            sum += num[k];
        }
        out.println(sum);
        out.close(); // don't forget this!
        System.exit(0);
    }
}
```

2 C++

Here I will use C++-style I/O. If you prefer C-style I/O, that's fine too.

2.1 Standard I/O

The first two lines here are to speed up input. They are considered bad coding practice outside of the contest environment but are essential to get your times down if I/O is large. Note, however, that if you unlink with C-style I/O, you may not use `scanf()` and `printf()`, etc. Bad things will happen.

```cpp
#include <iostream>
#include <fstream>

int num[100005];

int main() {
    std::ios_base::sync_with_stdio(0); // unlink C-style I/O
    std::cin.tie(0); // unlink std::cout
    std::cin >> N;
    for(int k = 0; k < N; ++k) {
        std::cin >> num[k];
    }
    int sum = 0;
    for(int k = 0; k < N; ++k) {
        sum += num[k];
    }
    cout << sum << "\n";
    return 0;
}
```

2.2 File I/O

```cpp
#include <iostream>
#include <fstream>

int num[100005];

int main() {
    std::ifstream fin("palpath.in");
    std::ofstream fout("palpath.out");
    fin >> N;
    for(int k = 0; k < N; ++k) {
        fin >> num[k];
    }
    int sum = 0;
    for(int k = 0; k < N; ++k) {
        sum += num[k];
    }
    fout << sum << "\n";
    fin.close();
    fout.close(); // don't forget this!
    return 0;
}
```

3 Python

3.1 Standard I/O

```python
n = int( input() ) # input() grabs the whole line
nums = input().strip() # removes extra spaces at beginning and end, also \n
nums = nums.split() # splits at the spaces to turn it into an array of strings
nums = [int(stng) for stng in nums] #turn them into ints
print( sum( nums ) )
```

3.2 File I/O

```python
file = open('input.txt', 'r') # r for read
out = open('output.txt', 'w') # w for write

n = int(file.readline().strip()) # strip() isn't always necessary, but it's a good habit
nums = file.readline().strip()
nums = nums.split() # splits at the spaces to turn it into an array of strings
nums = [int(stng) for stng in nums] #turn them into ints
out.write( str( sum( nums ) ) + '\n' ) # str() and + are necessary because write() takes
    only a single string

file.close()
out.close()
```