First Code Sprint – Sets

**Vasil Stanchev**

[VTStanchev18@codingburgas.bg](mailto:VTStanchev18@codingburgas.bg)

**Martin Sevov**

[MSSevov18@codingburgas.bg](mailto:EDKehayova@codingburgas.bg)

**Alexandra Laleva**

[ADLaleva18@codingburgas.bg](mailto:ADLaleva@codingburgas.bg)

**Boris Kanev**

[BBKanev18@codingburgas.bg](mailto:GAMihov18@codingburgas.bg)

**Georgi Mihov**

[GAMihov18@codingburgas.bg](mailto:GAMihov18@codingburgas.bg)

1. **Roles**

Vasil Stanchev Scrum Trainer

Martin Sevov Developer C++

Alexandra Laleva Front End

Boris Kanev Front End

Georgi Mihov Code Check

1. **About The Project**

We had to create a C++ program which represents the operations on sets realized as arrays where the user can define the number of elements in the array, as the elements have values you choose.

1. **Applications used**

We used Visual Studio 2019 and Github to make and work on the project. It was developed in C++

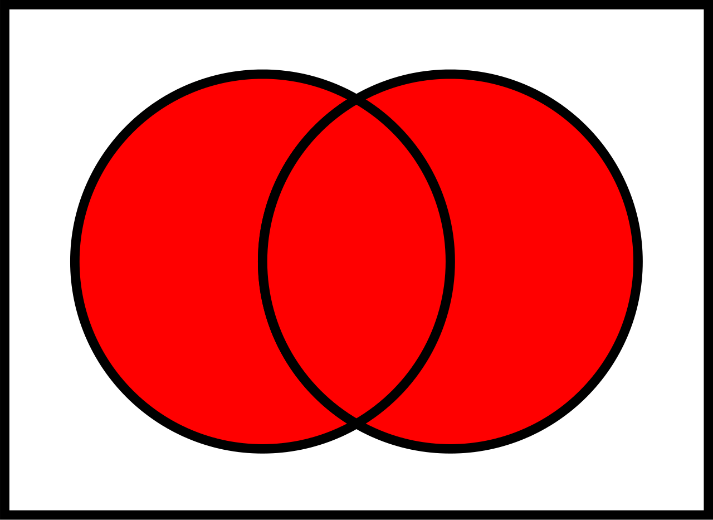
1. **Definition of Sets**

A set is a well-defined collection of distinct objects.

There are a few special sets:

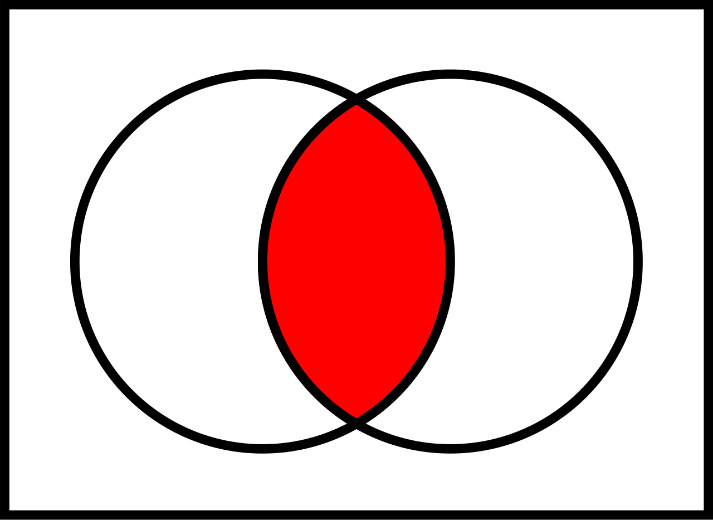
* P, denoting the set of all prime numbers: {2, 3, 5, …}
* N, denoting the set of all natural numbers: {0, 1, 2, …}
* Z, denoting the set of all integers: {…, -1, 0, 1, …}
* And a few more…

There are several fundamental operations for constructing new sets from given sets.

1. **Unions** - Two sets can be “*added*” together. The union of A and B, denoted by A ∪ B, is the set of all objects that are members of either A or B

The union of A and B

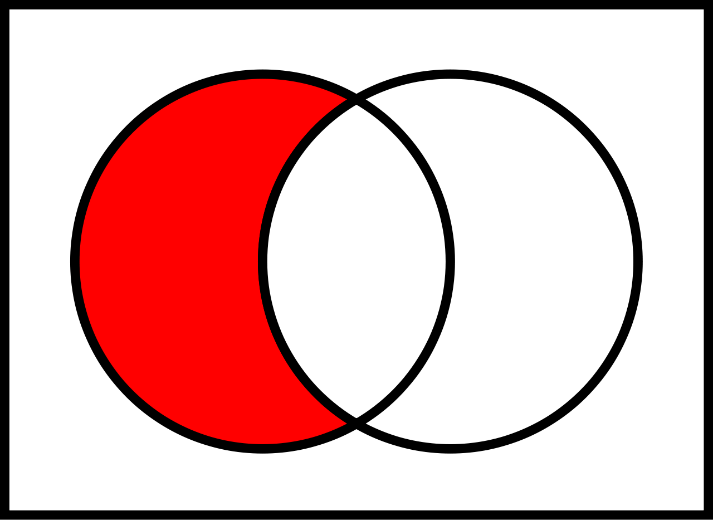
A ∪ B

1. **Intersections** - A new set is constructed by determining which members two sets have "*in common*". The intersection of A and B, denoted by A ∩ B, is the set of all things that are members of both A and B. If A ∩ B = ∅, then A and B are said to be disjoint.

The intersection of A and B

A ∩ B

1. **Complements** – Two sets can also be “*subtracted*”. The relative complement of B in A, denoted by A \ B



The relative complement of B in A

A \ B

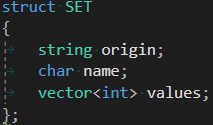
1. **About the Code**

The code is separated into two files: *Source.cpp*, *Sets.h*

Source.cpp’s purpose is to show off the functionality of the Sets header file.

In Source.cpp is the main menu that connects the functions from Sets.h.

There are a few Easter eggs, made for fun ☺.

 Sets.h is the main goal of the project. The header file consists of:

* A set structure with a name, origin and values
* Functions that create, delete and print sets
* Functions that check if a given value is allowed – used to increase stability
* Functions that create a new set through a fundamental operation

In the future we plan to add functions that save and load sets.