# Software Design and Implementation Seminar C

This exercise abstracts the collection and initial processing of command line arguments into one or more functions, which are intended as utilities that can easily be used in other programs. The functions need to deal with only one style of arguments and the parameters for each of them are exactly those passed to main()

### Function 1

```
sdi::vector<std::string> SDI::parseArgs(int argc, char * argv[] );
```

This function converts each of the parameters to a std::string and collects them all in a vector in the same order as argv; this vector is the return value of the function.

### Function 2

```
sdi::set<char> SDI::parseArgsFlags(int argc, char * argv[] );
```

This variant collects single character flags that will be set with perhaps a '-' or a '/' on the command line; you are free to impose restrictions as appropriate, such as each flag must have its own '-' or all of them must be in a no-space string. Case sensitivity is a consideration.

### Function 3

```
sdi::map<char, std::string> SDI::parseArgsValues(int argc,
                                                 char * argv[] );
```

This variant collects single character flags that are passed with values, such as

foo -x=12 -y=true -z=Simon

returning {('x', "12") , ('y', "true") , ('z', "Simon")}

or

bar -a=no -s=Richard -p=88356

returning {('a', "no") , ('p', "88356") , ('s', "Richard")}