CT103 Programming

Semester 2 Week 9

Command-Line Arguments

and Fun with Calendars

Dr. Sam Redfern

Discord Server (the same one we're using for CT1114)

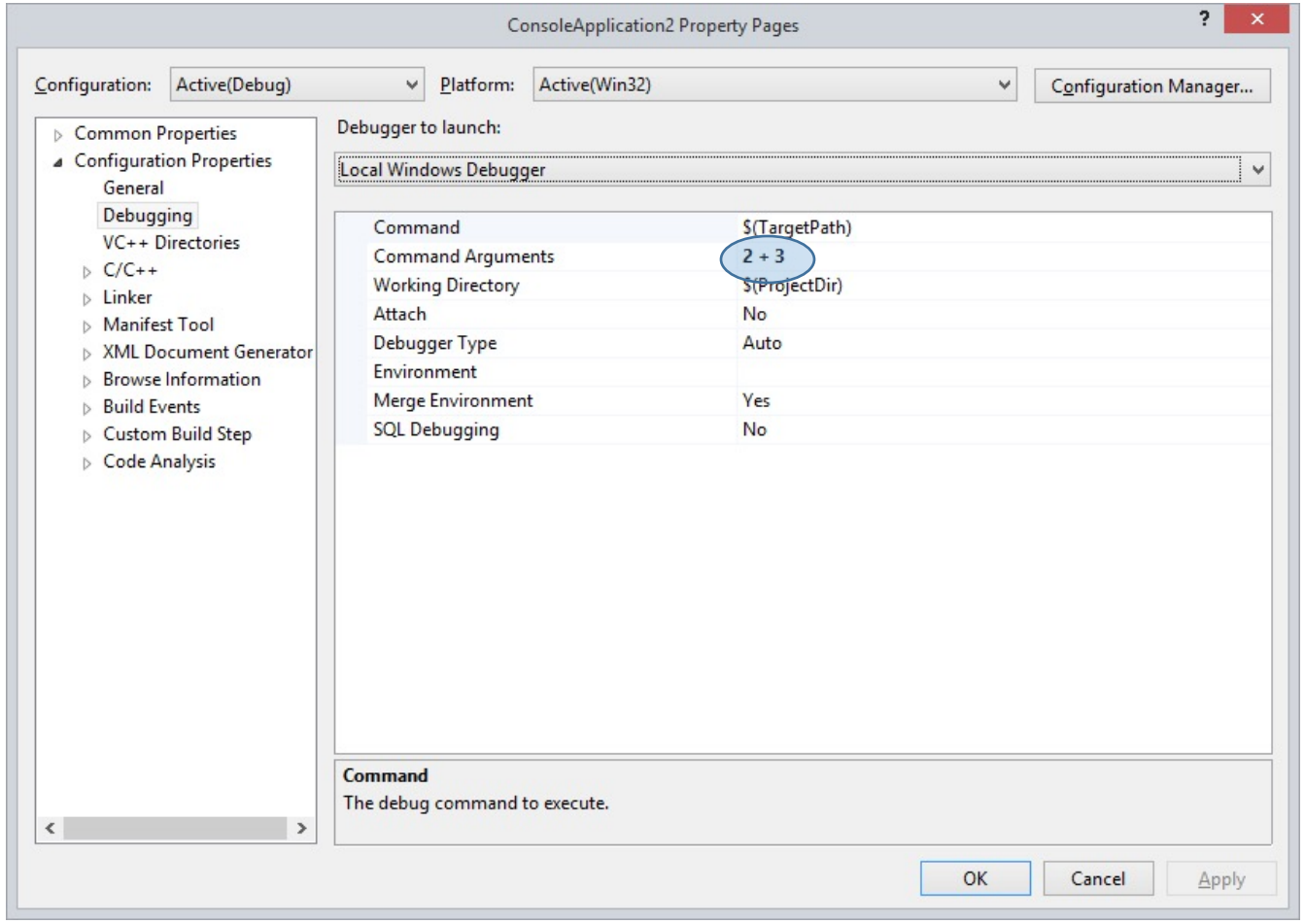
Recall from last week: names with cumulative frequencies

- Zachary Dartaghan is as common as Mary Smith??
- What is the cumulative frequency data and how we can use it..?
- Example: updating the previous solution to use frequencies, and produce statistically believable sets of names:
  - 09_random_names_with_freq.cpp
- My solution uses linear search.. could it use binary search?

Exercise: get this working on your computer!

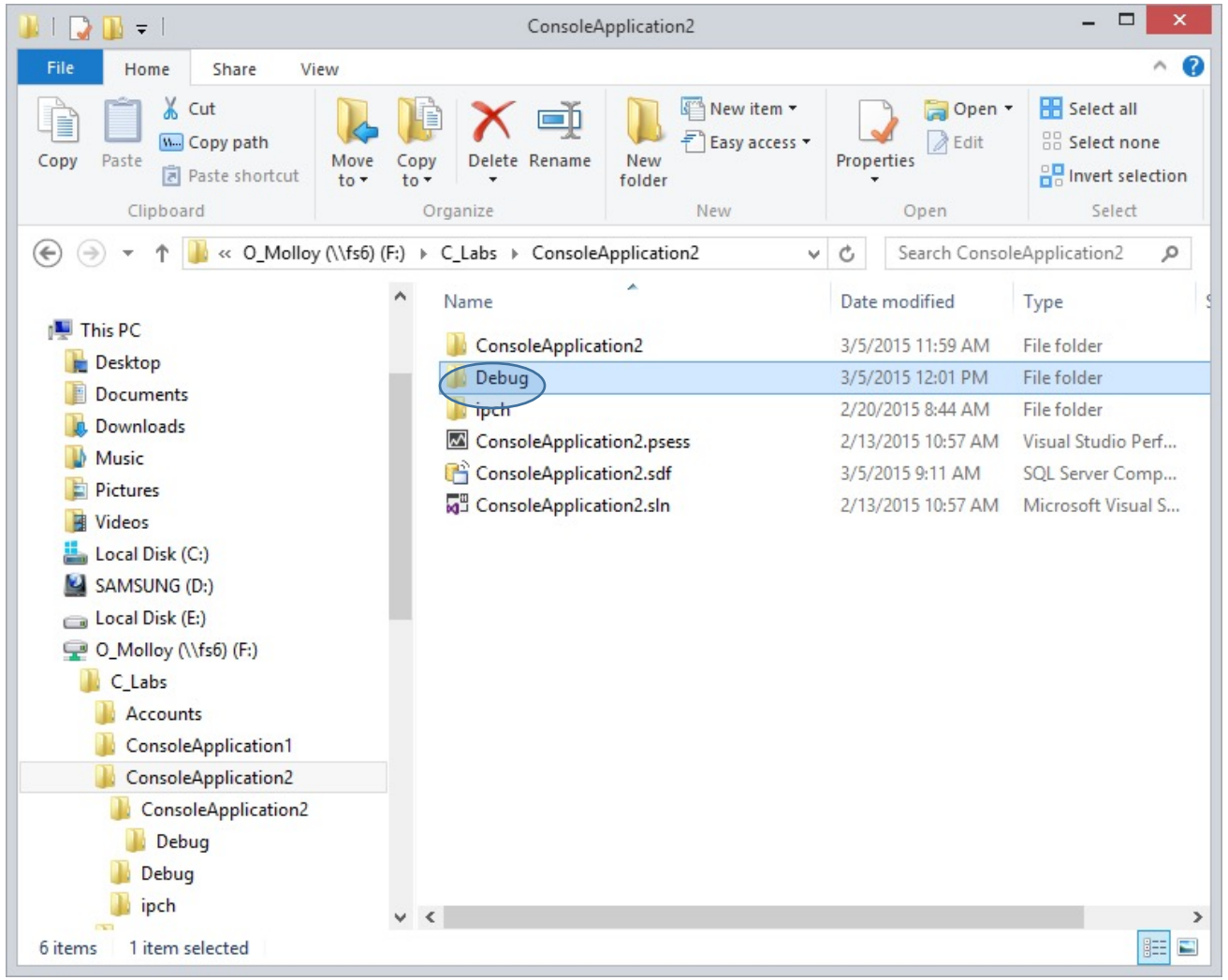
Command Line Arguments

- You can set the inputs (command line arguments) for your .exe in the option
- Project Properties
  - Configuration Properties
    - Debugging
      - Command Arguments

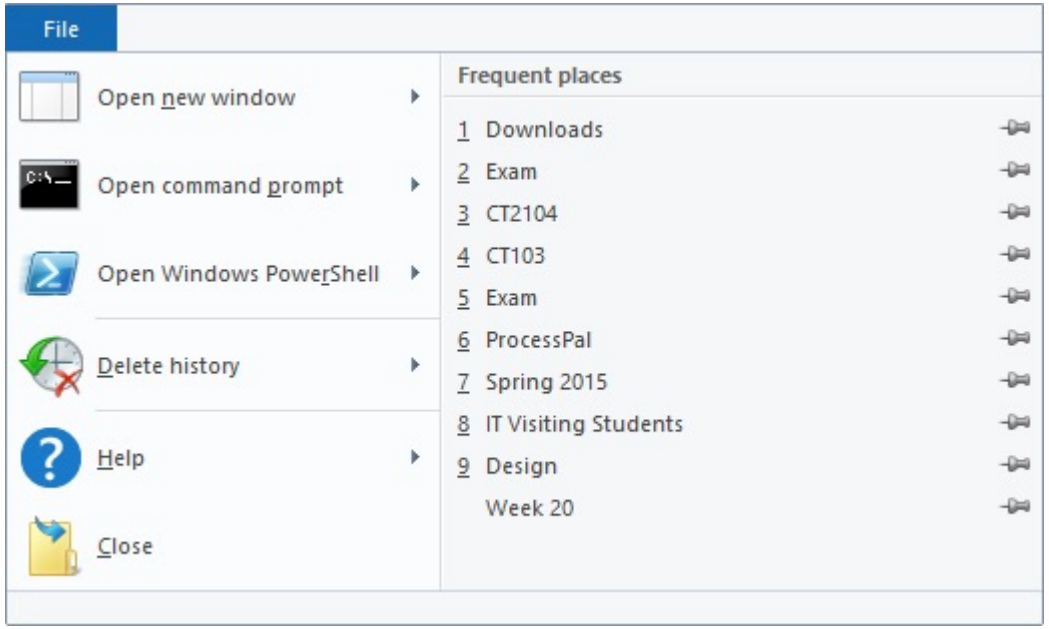


Or run from command line

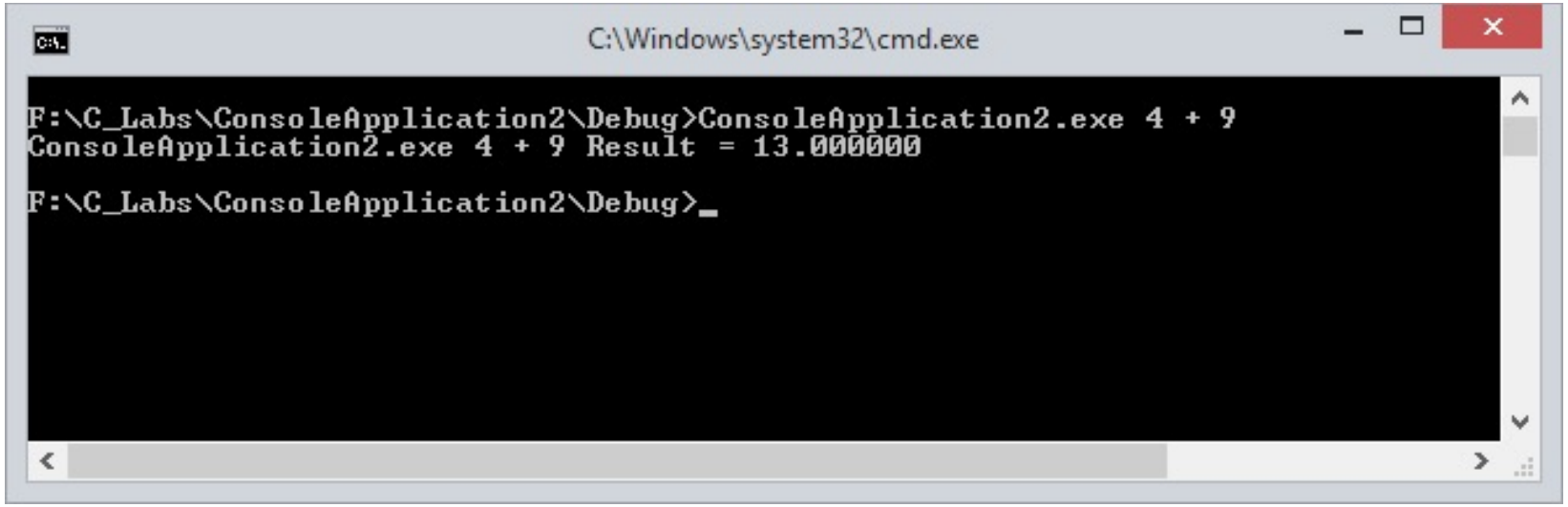
- Open the folder containing your solution
- There should be a Debug folder – open it
- It should contain the .exe



- File -> Open Command Prompt in this Debug folder



- Now you can run the .exe, typing in the full name of the executable, followed by the arguments

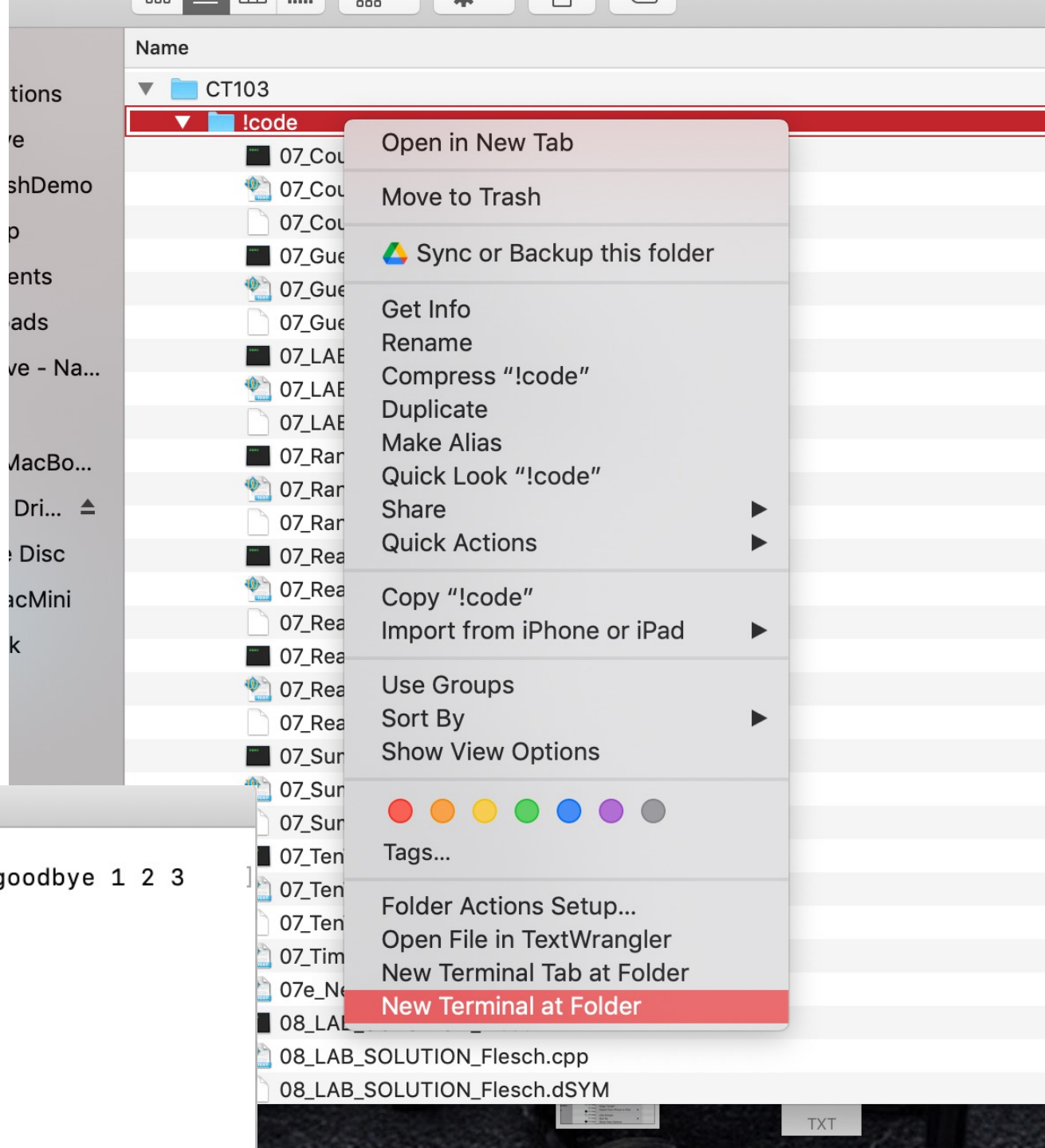


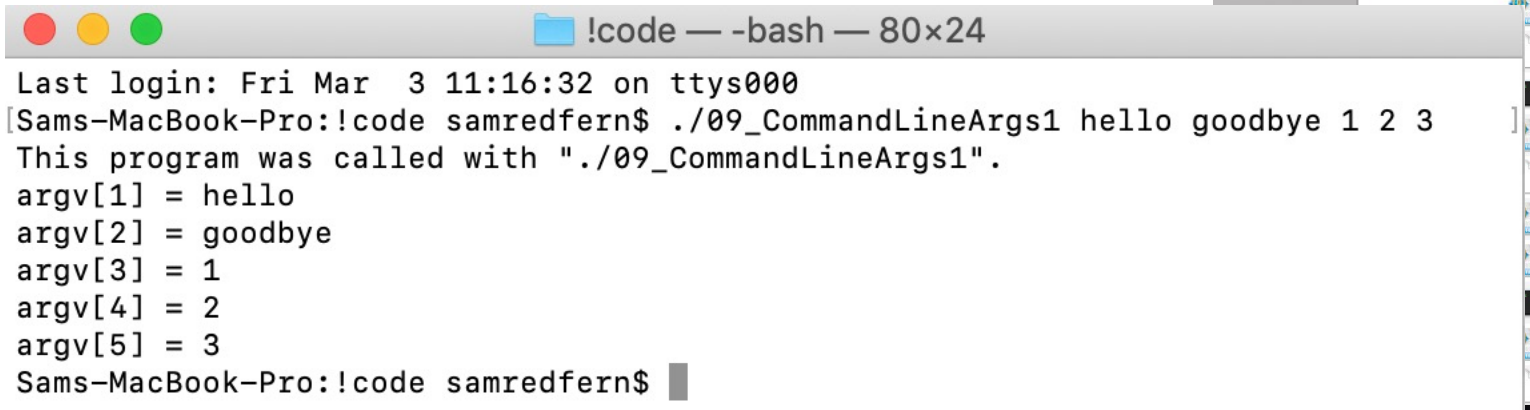
On the Mac (in XCode)

- In Xcode
- Product
  - Scheme
    - Edit Scheme
      - Run
        - Arguments
          - + (add arguments)

On the Mac (command line / terminal)

- Right click the folder where your executable was built
- “New Terminal at Folder”
- On the Mac you must put “./” before the program name to run it





Command line arguments example

```c
#include <stdio.h>

int main (int argc, char *argv[]) {
  int count;

  // argc is the number of command-line args (including exe name)
  // argv[0] is the exe name (including path)
  printf ("This program was called with \"%s\".\n", argv[0]);

  if (argc > 1) {
    // argv[1], argv[2] etc. are the "actual" arguments
    for (count = 1; count < argc; count++)
      printf("argv[%d] = %s\n", count, argv[count]);
    }
    else {
      printf("Called with no command-line arguments.\n");
    }

  return 0;
}
```

Exercise: get this working on your computer!

Exercise

- Take the *statistically-correct random names* program above (09_random_names_with_freq.cpp) and make it command-line driven
- The user should be able to specify using command-line arguments (i) the number of female names and (ii) the number of male names they want to have generated.

Exercise

- Add together all of the numbers supplied at the command-line
- You can convert a string to a number using atoi() or atof() from <stdlib.h>
  - **`int atoi(const char *str)`**
  - **`double atof(const char *str)`**

Some Exercises with Calendars

A function that returns the number of days in a month

```
int no_of_days(int year, int month) {
    if (month == 9 || month == 4 || month == 6 || month
== 11)
        return 30;

    if (month != 2)
        return 31;

    // but what about February?
}
```

Exercise: Leap Years

- Write a C function which receives a year number as an argument.
- The function should return 1 if the year is a leap year, and return 0 if it is not.
- Start with the code provided on the next slide

```c
#include <stdio.h>

int is_leap(int year);
int no_of_days(int year, int month);

int main() {
  int y;
  printf("Enter a year number > ");
  scanf(" %d", &y);
  if (is_leap(y)==1)
    printf("It's a leap year!");
  else
    printf("It's not a leap year!");
  printf(" ... and February has %d days.",
                          no_of_days(y,2));
}
```

```c
int no_of_days(int year, int month) {
  if (month == 9 || month == 4 || month
== 6 || month == 11)
    return 30;

  if (month != 2)
    return 31;

  return 28 + is_leap(year);
}


int is_leap(int year) {
  return 1;
  // to do: change this so that leap
years return 1
  // and others return 0
}
```

What day of the week does a month start on?

- Fact: January 1st, 1900 was a Monday

- So, what day of the week was February 1st, 1900?
  - And how do we calculate that?
- What day of the week was May 1st, 1900?
  - And how do we calculate that?
- What about May 1st, 1901?

Assignment

- Write a command-line-driven program which accepts a year number as an argument, and one or more month numbers as subsequent arguments
- The program should print out calendars for the specified months:
  - argv[1] = the year number
  - argv[2], argv[3] etc. are the month numbers to produce calendars for in that year. There could be just one month number, or there could be several.
- See starting code:

09_LAB_CommandLineCalendarsStart.cpp

```
.\calendars.exe 2021 3 4


3/2021

 Sun Mon Tue Wed Thu Fri Sat
       1   2   3   4   5   6
   7   8   9  10  11  12  13
  14  15  16  17  18  19  20
  21  22  23  24  25  26  27
  28  29  30  31

4/2021

 Sun Mon Tue Wed Thu Fri Sat
                     1   2   3
   4   5   6   7   8   9  10
  11  12  13  14  15  16  17
  18  19  20  21  22  23  24
  25  26  27  28  29  30
```