Algorithms and Analysis

Lesson 4: *C++ 101*

IN CONGRESS,
JULY 4, 1776.

A DECLARATION BY THE REPRESENTATIVES OF THE UNITED STATES OF AMERICA, IN GENERAL CONGRESS ASSEMBLED.

WHEN in the course of human events, it becomes necessary for one people to dissolve the political bands which have connected them with another, and to assume among the powers of the earth, the separate and equal station to which the laws of nature and of nature's God entitle them, a decent respect to the opinions of mankind requires that they should declare the causes which impel them to the separation.

We hold these truths to be self-evident, that all men are created equal, that they are endowed by their Creator with certain unalienable rights, that among these are life, liberty, and the pursuit of happiness. That to secure these rights, governments are instituted among men, deriving their just powers from the consent of the governed, that whenever any form of government becomes destructive of these ends, it is the right of the people to alter or to abolish it, and to institute new government, laying its foundation on such principles, and organizing its powers in such form, as to them shall seem most likely to effect their safety and happiness. Prudence, indeed, will dictate that governments long established should not be changed for light and transient causes; and accordingly all experience hath shewn, that mankind are more disposed to suffer, while evils are sufferable, than to right themselves by abolishing the forms to which they are accustomed. But when a long train of abuses and usurpations, pursuing invariably the same object, evinces a design to reduce them under absolute despotism, it is their right, it is their duty, to throw off such government, and to provide new guards for their future security. Such has been the patient sufferance of these Colonies; and such is now the necessity which constrains them to alter their former systems of government. The history of the present King of Great-Britain is a history of repeated injuries and usurpations, all having in direct object the establishment of an absolute tyranny over these States. To prove this, let facts be submitted to a candid world.

He has refused his assent to laws, the most wholesome and necessary for the public good.

He has forbidden his Governors to pass laws of immediate and pressing importance, unless suspended in their operation till his assent should be obtained; and when so suspended, he has utterly neglected to attend them.

He has refused to pass other laws for the accommodation of large districts of people, unless those people would relinquish the right of representation in the legislature, a right inestimable to them, and formidable to Tyrants only.

He has called together legislative bodies at places unusual, uncomfortable, and distant from the depository of their public records, for the sole purpose of fatiguing them into compliance with his measures.

He has dissolved representative houses repeatedly, for opposing with manly firmness his invasions on the rights of the people.

He has refused for a long time, after such dissolutions, to cause others to be elected; whereby the legislative powers, incapable of annihilation, have returned to the people at large for their exercise; the State remaining in the mean time exposed to all the dangers of invasion from without, and convulsions within.

He has endeavored to prevent the population of these States; for that purpose obstructing the laws for naturalization of foreigners; refusing to pass others to encourage their migrations hither, and raising the conditions of new appropriations of lands.

He has obstructed the administration of justice, by refusing his assent to laws for establishing judiciary powers.

He has made Judges dependent on his will alone, for the tenure of their offices, and the amount and payment of their salaries.

He has erected a multitude of new offices, and sent hither swarms of officers to harass our people, and eat out their substance.

He has kept among us, in times of peace, standing armies, without the consent of our Legislatures.

He has affected to render the military independent of and superior to the civil power.

He has combined with others to subject us to a jurisdiction foreign to our constitution, and unacknowledged by our laws; giving his assent to their acts of pretended legislation:

For quartering large bodies of armed troops among us:

For protecting them, by a mock trial, from punishment for any murders which they should commit on the inhabitants of these States:

For cutting off our trade with all parts of the world:

For imposing taxes on us without our consent:

For depriving us, in many cases, of the benefits of trial by jury:

For transporting us beyond seas to be tried for pretended offenses:

For abolishing the free system of English laws in a neighboring province, establishing therein an arbitrary government, and enlarging its boundaries, so as to render it at once an example and fit instrument for introducing the same absolute rule into these Colonies:

For taking away our Charters, abolishing our most valuable laws, and altering fundamentally the forms of our governments:

For suspending our own Legislatures, and declaring themselves invested with power to legislate for us in all cases whatsoever.

He has abdicated government here, by declaring us out of his protection and waging war against us.

He has plundered our seas, ravaged our coasts, burnt our towns, and destroyed the lives of our people.

He is at this time, transporting large armies of foreign mercenaries to compleat the works of death, desolation and tyranny, already begun with circumstances of cruelty and perfidy scarcely paralleled in the most barbarous ages, and totally unworthy the head of a civilized nation.

He has constrained our fellow citizens taken captive on the high seas to bear arms against their country, to become the executioners of their friends and brethren, or to fall themselves by their hands.

He has excited domestic insurrections amongst us, and has endeavored to bring on the inhabitants of our frontiers, the merciless Indian Savages, whose known rule of warfare, is an undistinguished destruction of all ages, sexes, and conditions.

In every stage of these oppressions we have petitioned for redress, in the most humble terms: Our repeated petitions have been answered only by repeated injury. A Prince, whose character is thus marked by every act which may define a Tyrant, is unfit to be the Ruler of a free People.

Nor have we been wanting in attentions to our British brethren. We have warned them from time to time of attempts by their legislature to extend an unwarrantable jurisdiction over us. We have reminded them of the circumstances of our emigration and settlement here. We have appealed to their native justice and magnanimity, and we have conjured them by the ties of our common kindred to disavow these usurpations, which would inevitably interrupt our connections and correspondence. They too have been deaf to the voice of justice and of consanguinity. We must, therefore, acquiesce in the necessity which denounces our separation, and hold them, as we hold the rest of mankind, enemies in war, in peace, friends.

We, therefore, the Representatives of the UNITED STATES OF AMERICA, in GENERAL CONGRESS assembled, appealing to the Supreme Judge of the world for the rectitude of our intentions, do in the name and by the authority of the good People of these Colonies, solemnly publish and declare, That these United Colonies are, and of right ought to be, FREE AND INDEPENDENT STATES; that they are absolved from all allegiance to the British Crown, and that all political connection between them and the state of Great-Britain, is, and ought to be totally dissolved; and that as FREE AND INDEPENDENT STATES, they have full power to levy war, conclude peace, contract alliances, establish commerce, and do all other acts and things which INDEPENDENT STATES may of right do. And for the support of this Declaration, with a firm reliance on the protection of Divine Providence, we mutually pledge to each other our lives, our fortunes, and our sacred honor.

Signed by ORDER *and in* BEHALF *of the* CONGRESS,
JOHN HANCOCK, PRESIDENT.
ATTEST,
CHARLES THOMPSON, Secretary.

C with classes, new, overloading, templates

Outline

1. **C with Classes**
2. New
3. Overloading
4. Templates

IN

CONGRESS,

JULY 4, 1776.

A

DECLARATION

BY THE

REPRESENTATIVES

OF THE

UNITED STATES OF AMERICA,

In GENERAL CONGRESS assembled.

WHEN in the course of human events, it becomes necessary for one people to dissolve the political bands which have connected them with another, and to assume among the powers of the earth, the separate and equal station to which the laws of nature and of nature's God entitle them, a decent respect to the opinions of mankind requires that they should declare the causes which impel them to the separation.

We hold these truths to be self-evident, that all men are created equal, that they are endowed by their Creator with certain unalienable rights, that among these are life, Liberty, and the pursuit of happiness. That to secure these rights, governments are instituted among men, deriving their just powers from the consent of the governed, that whenever any form of government becomes destructive of these ends, it is the right of the people to alter or to abolish it, and to institute new government, laying its foundation on such principles, and organizing its powers in such form, as to them shall seem most likely to effect their safety and happiness. Prudence, indeed, will dictate that governments long established should not be changed for light and transient causes; and accordingly all experience hath shewn, that mankind are more disposed to suffer, while evils are sufferable, than to right themselves by abolishing the forms to which they are accustomed. But when a long train of abuses and usurpations, pursuing invariably the same object, evinces a design to reduce them under absolute despotism, it is their right, it is their duty, to throw off such government, and to provide new guards for their future security. Such has been the patient sufferance of these Colonies; and such is now the necessity which constrains them to alter their former systems of government. The history of the present King of Great-Britain is a history of repeated injuries and usurpations, all having in direct object the establishment of an absolute tyranny over these states. To prove this, let facts be submitted to a candid world.

He has refused his assent to laws, the most wholesome and necessary for the public good.

He has forbidden his Governors to pass laws of immediate and pressing importance, unless suspended in their operation till his assent should be obtained; and when so suspended, he has utterly neglected to attend them.

He has refused to pass other laws for the accommodation of large districts of people, unless those people would relinquish the right of representation in the legislature, a right inestimable to them, and formidable to Tyrants only.

He has called together legislative bodies at places unusual, uncomfortable, and distant from the depository of their public records, for the sole purpose of fatiguing them into compliance with his measures.

He has dissolved representative houses repeatedly, for opposing with manly firmness his invasions on the rights of the people.

He has refused for a long time, after such dissolutions, to cause others to be elected; whereby the legislative powers, incapable of annihilation, have returned to the people at large for their exercise; the state remaining in the mean time exposed to all the dangers of invasion from without, and convulsions within.

He has endeavored to prevent the population of these States; for that purpose obstructing the laws for naturalization of foreigners; refusing to pass others to encourage their migrations hither, and raising the conditions of new appropriations of lands.

He has obstructed the administration of justice, by refusing his assent to laws for establishing judiciary powers.

He has made Judges dependent on his will alone, for the tenure of their offices, and the amount and payment of their salaries.

He has erected a multitude of new offices, and sent hither swarms of officers to harass our people, and eat out their substance.

He has kept among us, in times of peace, standing armies, without the consent of our Legislatures.

He has affected to render the military independent of and superior to the civil power.

He has combined with others to subject us to a jurisdiction foreign to our constitution, and unacknowledged by our laws; giving his assent to their acts of pretended legislation:

For quartering large bodies of armed troops among us:

For protecting them, by a mock trial, from punishment for any murders which they should commit on the inhabitants of these States:

For cutting off our trade with all parts of the world:

For imposing taxes on us without our consent:

For depriving us, in many cases, of the benefits of trial by jury:

For transporting us beyond seas to be tried for pretended offenses:

For abolishing the free system of English laws in a neighboring province, establishing therein an arbitrary government, and enlarging its boundaries, so as to render it at once an example and fit instrument for introducing the same absolute rule into these Colonies:

For taking away our Charters, abolishing our most valuable laws, and altering fundamentally the forms of our governments:

For suspending our own Legislatures, and declaring themselves invested with power to legislate for us in all cases whatsoever.

He has abdicated government here, by declaring us out of his protection and waging war against us.

He has plundered our seas, ravaged our coasts, burnt our towns, and destroyed the lives of our people.

He is at this time, transporting large armies of foreign mercenaries to compleat the works of death, desolation and tyranny, already begun with circumstances of cruelty and perfidy scarcely paralleled in the most barbarous ages, and totally unworthy the head of a civilized nation.

He has constrained our fellow citizens taken captive on the high seas to bear arms against their country, to become the executioners of their friends and brethren, or to fall themselves by their hands.

He has excited domestic insurrections amongst us, and has endeavored to bring on the inhabitants of our frontiers, the merciless Indian Savages, whose known rule of warfare, is an undistinguished destruction of all ages, sexes, and conditions.

In every stage of these oppressions we have petitioned for redress in the most humble terms: Our repeated petitions have been answered only by repeated injury. A Prince, whose character is thus marked by every act which may define a Tyrant, is unfit to be the Ruler of a free People.

Nor have we been wanting in attentions to our British brethren. We have warned them from time to time of attempts by their legislature to extend an unwarrantable jurisdiction over us. We have reminded them of the circumstances of our emigration and settlement here. We have appealed to their native justice and magnanimity, and we have conjured them by the ties of our common kindred to disavow these usurpations, which would inevitably interrupt our connections and correspondence. They too have been deaf to the voice of justice and of consanguinity. We must, therefore, acquiesce in the necessity which denounces our separation, and hold them, as we hold the rest of mankind, enemies in war; in peace, friends.

We, therefore, the Representatives of the UNITED STATES OF AMERICA, in GENERAL CONGRESS assembled, appealing to the Supreme Judge of the world for the rectitude of our intentions, do in the name and by the authority of the good People of these Colonies, solemnly publish and declare, That these United Colonies are, and of right ought to be, FREE AND INDEPENDENT STATES; that they are absolved from all allegiance to the British Crown, and that all political connection between them and the State of Great-Britain, is, and ought to be totally dissolved; and that as FREE AND INDEPENDENT STATES, they have full power to levy war, conclude peace, contract alliances, establish commerce, and to do all other acts and things which INDEPENDENT STATES may of right do. And for the support of this Declaration, with a firm reliance on the protection of Divine Providence, we mutually pledge to each other our lives, our fortunes, and our sacred honor.

Signed by ORDER *and in* BEHALF *of the* CONGRESS,

JOHN HANCOCK, PRESIDENT.

ATTEST,

CHARLES THOMPSON, Secretary.

C

- C was developed in the 1970s by Dennis Ritchie for writing UNIX tools
- It supported structural programming through functions
- It allowed run-time allocation of memory (through `malloc` and `free`)
- It allowed manipulation of memory through pointers
- This made it efficient

C

- C was developed in the 1970s by Dennis Ritchie for writing UNIX tools
- It supported structural programming through functions
- It allowed run-time allocation of memory (through `malloc` and `free`)
- It allowed manipulation of memory through pointers
- This made it efficient

C

- C was developed in the 1970s by Dennis Ritchie for writing UNIX tools
- It supported structural programming through functions
- It allowed run-time allocation of memory (through `malloc` and `free`)
- It allowed manipulation of memory through pointers
- This made it efficient

C

- C was developed in the 1970s by Dennis Ritchie for writing UNIX tools
- It supported structural programming through functions
- It allowed run-time allocation of memory (through `malloc` and `free`)
- It allowed manipulation of memory through pointers
- This made it efficient

C

- C was developed in the 1970s by Dennis Ritchie for writing UNIX tools
- It supported structural programming through functions
- It allowed run-time allocation of memory (through `malloc` and `free`)
- It allowed manipulation of memory through pointers
- This made it efficient

C

- C was developed in the 1970s by Dennis Ritchie for writing UNIX tools
- It supported structural programming through functions
- It allowed run-time allocation of memory (through `malloc` and `free`)
- It allowed manipulation of memory through pointers
- This made it efficient, but not safe

C

- C was developed in the 1970s by Dennis Ritchie for writing UNIX tools
- It supported structural programming through functions
- It allowed run-time allocation of memory (through `malloc` and `free`)
- It allowed manipulation of memory through pointers
- This made it efficient, but not safe or easy to use

Keeping Things Together

- As soon as you start programming bigger systems you want to keep information together

- C facilitated this through C structures `struct`

```
struct MyStructure {    // Structure declaration
  int myNum;            // Member (int variable)
  char myLetter;        // Member (char variable)
}; // End the structure with a semicolon

int main() {
  struct myStructure s1;

  s1.myNum = 13;
  s1.myLetter = 'B';

  printf("My number: %d\n", s1.myNum);
  printf("My letter: %c\n", s1.myLetter);
  return 0;
}
```

Estimated Errors in the Mean

- When working with empirical data, X_i we want to compute the $mean$ and $variance$ (from which we can estimate the error in the mean)

- We can do this on the fly by storing

$$n, \qquad \hat{\mu}_n = \frac{1}{n}\sum_{i=1}^{n} X_i, \qquad Q_n = \sum_{i=1}^{n}(X_i - \hat{\mu}_n).$$

- Given X_{n+1} we can update our data using

$$\Delta = \frac{X_{n+1} - \hat{\mu}_n}{n+1},\ Q_{n+1} = Q_n + n\,\Delta\,(X_{n+1} - \hat{\mu}_n),\ \hat{\mu}_{n+1} = \hat{\mu}_n + \Delta.$$

  this requires the back of an envelop to verify

Estimated Errors in the Mean

- When working with empirical data, X_i we want to compute the $mean$ and $variance$ (from which we can estimate the error in the mean)

- We can do this on the fly by storing

$$n, \qquad \hat{\mu}_n = \frac{1}{n}\sum_{i=1}^{n} X_i, \qquad Q_n = \sum_{i=1}^{n}(X_i - \hat{\mu}_n).$$

- Given X_{n+1} we can update our data using

$$\Delta = \frac{X_{n+1} - \hat{\mu}_n}{n+1},\ Q_{n+1} = Q_n + n\,\Delta\,(X_{n+1} - \hat{\mu}_n),\ \hat{\mu}_{n+1} = \hat{\mu}_n + \Delta.$$

  this requires the back of an envelop to verify

Estimated Errors in the Mean

- When working with empirical data, X_i we want to compute the $mean$ and $variance$ (from which we can estimate the error in the mean)

- We can do this on the fly by storing

$$n, \qquad \hat{\mu}_n = \frac{1}{n}\sum_{i=1}^{n} X_i, \qquad Q_n = \sum_{i=1}^{n}(X_i - \hat{\mu}_n).$$

- Given X_{n+1} we can update our data using

$$\Delta = \frac{X_{n+1} - \hat{\mu}_n}{n+1},\ Q_{n+1} = Q_n + n\,\Delta\,(X_{n+1} - \hat{\mu}_n),\ \hat{\mu}_{n+1} = \hat{\mu}_n + \Delta.$$

this requires the back of an envelop to verify

Second Order Statistics in C

- In C we can use a `struct` to keep this data together

```
struct Sos {
  unsigned n;
  double mu;
  double Q;
};
```

- We can write functions that update thos

```
void add(struct Sos& sos, x) {
  double delta = = (x - mu)/(n+1.0);
  Q += n*delta*(x - mu);
  n += delta;
  n++;
}
```

Second Order Statistics in C

- In C we can use a `struct` to keep this data together

```
struct Sos {
  unsigned n;
  double mu;
  double Q;
};
```

- We can write functions that update thos

```
void add(struct Sos& sos, x) {
  double delta = = (x - mu)/(n+1.0);
  Q += n*delta*(x - mu);
  n += delta;
  n++;
}
```

Classes

- C++ was developed by Bjarne Stroustrup and released in 1985 as "C with classes"
- It was syntactic sugar that compiled down to C (as such if was intended to be as fast as C)
- You are familiar with classes from python and they are very much the same thing

Classes

- C++ was developed by Bjarne Stroustrup and released in 1985 as "C with classes"
- It was syntactic sugar that compiled down to C (as such if was intended to be as fast as C)
- You are familiar with classes from python and they are very much the same thing

Classes

- C++ was developed by Bjarne Stroustrup and released in 1985 as "C with classes"
- It was syntactic sugar that compiled down to C (as such if was intended to be as fast as C)
- You are familiar with classes from python and they are very much the same thing

Classes

- C++ was developed by Bjarne Stroustrup and released in 1985 as "C with classes"
- It was syntactic sugar that compiled down to C (as such if was intended to be as fast as C)
- You are familiar with classes from python and they are very much the same thing

Classes

- C++ was developed by Bjarne Stroustrup and released in 1985 as "C with classes"
- It was syntactic sugar that compiled down to C (as such if was intended to be as fast as C)
- You are familiar with classes from python and they are very much the same thing, except C++ is a lot more elegant than python

Classes

- C++ was developed by Bjarne Stroustrup and released in 1985 as "C with classes"
- It was syntactic sugar that compiled down to C (as such if was intended to be as fast as C)
- You are familiar with classes from python and they are very much the same thing, except C++ is a lot more elegant than python
- It has grown since 1985, adding templates and a lot of nice functionality

Classes by Example

- Define programme in header file `sos.h`

```
class Sos {
private:                    // encapsulate
  int n;
  double mu;
  double Q;

public:                     // interface
  Sos();                    // constructor
  void add(double x);       // add data
  double mean();            // return mean
  double var();             // unbiased estimate of variance
  double error();           // estimated error in mean
}
```

Classes by Example

- Define programme in header file `sos.h`

```
class Sos {
private:                    // encapsulate
  int n;
  double mu;
  double Q;

public:                     // interface
  Sos();                    // constructor
  void add(double x);       // add data
  double mean();            // return mean
  double var();             // unbiased estimate of variance
  double error();           // estimated error in mean
}
```

Classes by Example

- Define programme in header file `sos.h`

```
class Sos {
private:                    // encapsulate
  int n;
  double mu;
  double Q;

public:                     // interface
  Sos();                    // constructor
  void add(double x);       // add data
  double mean();            // return mean
  double var();             // unbiased estimate of variance
  double error();           // estimated error in mean
}
```

Classes by Example

- Define programme in header file `sos.h`

```
class Sos {
private:                    // encapsulate
  int n;
  double mu;
  double Q;

public:                     // interface
  Sos();                    // constructor
  void add(double x);       // add data
  double mean();            // return mean
  double var();             // unbiased estimate of variance
  double error();           // estimated error in mean
}
```

Classes by Example

- Define programme in header file `sos.h`

```
class Sos {
private:                    // encapsulate
  int n;
  double mu;
  double Q;

public:                     // interface
  Sos();                    // constructor
  void add(double x);       // add data
  double mean();            // return mean
  double var();             // unbiased estimate of variance
  double error();           // estimated error in mean
}
```

Classes by Example

- Define programme in header file `sos.h`

```
class Sos {
private:                    // encapsulate
  int n;
  double mu;
  double Q;

public:                     // interface
  Sos();                    // constructor
  void add(double x);       // add data
  double mean();            // return mean
  double var();             // unbiased estimate of variance
  double error();           // estimated error in mean
}
```

Classes by Example

- Define programme in header file `sos.h`

```
class Sos {
private:                    // encapsulate
  int n;
  double mu;
  double Q;

public:                     // interface
  Sos();                    // constructor
  void add(double x);       // add data
  double mean();            // return mean
  double var();             // unbiased estimate of variance
  double error();           // estimated error in mean
}
```

Implementation of sos.cc

```
Sos::Sos() {n=0; mu=0.0; Q=0.0;}

void Sos::add(struct Sos& sos, x) {
  double delta = = (x - mu)/(n+1.0);
  Q += n*delta*(x - mu);
  n += delta;
  n++;
}

double Sos::mean() const {return mu;}

double Sos::var() const
{
  assert(n>1.0);
  return nvar/(n-1.0);
}

double error() const
{
  sqrt(var()/n);
}
```

Implementation of sos.cc

```
Sos::Sos() {n=0; mu=0.0; Q=0.0;}

void Sos::add(struct Sos& sos, x) {
  double delta = = (x - mu)/(n+1.0);
  Q += n*delta*(x - mu);
  n += delta;
  n++;
}

double Sos::mean() const {return mu;}

double Sos::var() const
{
  assert(n>1.0);
  return nvar/(n-1.0);
}

double error() const
{
  sqrt(var()/n);
}
```

Implementation of sos.cc

```
Sos::Sos() {n=0; mu=0.0; Q=0.0;}

void Sos::add(struct Sos& sos, x) {
  double delta = = (x - mu)/(n+1.0);
  Q += n*delta*(x - mu);
  n += delta;
  n++;
}

double Sos::mean() const {return mu;}

double Sos::var() const
{
  assert(n>1.0);
  return nvar/(n-1.0);
}

double error() const
{
  sqrt(var()/n);
}
```

Implementation of sos.cc

```
Sos::Sos() {n=0; mu=0.0; Q=0.0;}

void Sos::add(struct Sos& sos, x) {
  double delta = = (x - mu)/(n+1.0);
  Q += n*delta*(x - mu);
  n += delta;
  n++;
}

double Sos::mean() const {return mu;}

double Sos::var() const
{
  assert(n>1.0);
  return nvar/(n-1.0);
}

double error() const
{
  sqrt(var()/n);
}
```

Implementation of sos.cc

```
Sos::Sos() {n=0; mu=0.0; Q=0.0;}

void Sos::add(struct Sos& sos, x) {
  double delta = = (x - mu)/(n+1.0);
  Q += n*delta*(x - mu);
  n += delta;
  n++;
}

double Sos::mean() const {return mu;}

double Sos::var() const
{
  assert(n>1.0);
  return nvar/(n-1.0);
}

double error() const
{
  sqrt(var()/n);
}
```

Using Classes

- Classes are super easy to use

```
#include "sos.h"
using namespace std;

void main() {
  Sos mean;
  for(int i=0; i<n; ++i) {
    // compute X
    mean += X;
  }
  cout << mean.mean() << ' ' << mean.error() << endl;
}
```

- `Sos` is the class that I use most (both in C++ and python)

Using Classes

- Classes are super easy to use

```
#include "sos.h"
using namespace std;

void main() {
  Sos mean;
  for(int i=0; i<n; ++i) {
    // compute X
    mean += X;
  }
  cout << mean.mean() << ' ' << mean.error() << endl;
}
```

- `Sos` is the class that I use most (both in C++ and python)

Using Classes

- Classes are super easy to use

```
#include "sos.h"
using namespace std;

void main() {
  Sos mean;
  for(int i=0; i<n; ++i) {
    // compute X
    mean += X;
  }
  cout << mean.mean() << ' ' << mean.error() << endl;
}
```

- `Sos` is the class that I use most (both in C++ and python)

Using Classes

- Classes are super easy to use

```
#include "sos.h"
using namespace std;

void main() {
  Sos mean;
  for(int i=0; i<n; ++i) {
    // compute X
    mean += X;
  }
  cout << mean.mean() << ' ' << mean.error() << endl;
}
```

- `Sos` is the class that I use most (both in C++ and python)

Using Classes

- Classes are super easy to use

```
#include "sos.h"
using namespace std;

void main() {
  Sos mean;
  for(int i=0; i<n; ++i) {
    // compute X
    mean += X;
  }
  cout << mean.mean() << ' ' << mean.error() << endl;
}
```

- `Sos` is the class that I use most (both in C++ and python)

Using Classes

- Classes are super easy to use

```
#include "sos.h"
using namespace std;

void main() {
  Sos mean;
  for(int i=0; i<n; ++i) {
    // compute X
    mean += X;
  }
  cout << mean.mean() << ' ' << mean.error() << endl;
}
```

- `Sos` is the class that I use most (both in C++ and python)

Libraries

- C++ comes with a lot of in built libraries
- I include libraries using include statements

```
#include <iostream>
#include <vector>
```

- This is the same as C, but the C++ libraries don't have ".h
- These are known as the standard library or the standard template library

Libraries

- C++ comes with a lot of in built libraries
- I include libraries using include statements

```
#include <iostream>
#include <vector>
```

- This is the same as C, but the C++ libraries don't have ".h
- These are known as the standard library or the standard template library

Libraries

- C++ comes with a lot of in built libraries
- I include libraries using include statements

```
#include <iostream>
#include <vector>
```

- This is the same as C, but the C++ libraries don't have ".h
- These are known as the standard library or the standard template library

Libraries

- C++ comes with a lot of in built libraries
- I include libraries using include statements

```
#include <iostream>
#include <vector>
```

- This is the same as C, but the C++ libraries don't have ".h
- These are known as the standard library or the standard template library

Namespaces

- When you are writing very large programmes (possibly involving other peoples code) you might accidentally use the same name for a class, function or variable used elsewhere
- If you are luck this won't compile, or crash. If you are unlucky you will have a weird bug that will be very difficult to find
- To prevent this, C++ invented a new scope called **namespaces**
- By default all the standard library classes and functions are in namespace `std`
- To call the library we write `std::vector<double>`
- We can be lazy and write `using namespace std;`

Namespaces

- When you are writing very large programmes (possibly involving other peoples code) you might accidentally use the same name for a class, function or variable used elsewhere
- If you are luck this won't compile, or crash. If you are unlucky you will have a weird bug that will be very difficult to find
- To prevent this, C++ invented a new scope called **namespaces**
- By default all the standard library classes and functions are in namespace `std`
- To call the library we write `std::vector<double>`
- We can be lazy and write `using namespace std;`

Namespaces

- When you are writing very large programmes (possibly involving other peoples code) you might accidentally use the same name for a class, function or variable used elsewhere
- If you are luck this won't compile, or crash. If you are unlucky you will have a weird bug that will be very difficult to find
- To prevent this, C++ invented a new scope called **namespaces**
- By default all the standard library classes and functions are in namespace `std`
- To call the library we write `std::vector<double>`
- We can be lazy and write `using namespace std;`

Namespaces

- When you are writing very large programmes (possibly involving other peoples code) you might accidentally use the same name for a class, function or variable used elsewhere
- If you are luck this won't compile, or crash. If you are unlucky you will have a weird bug that will be very difficult to find
- To prevent this, C++ invented a new scope called **namespaces**
- By default all the standard library classes and functions are in namespace `std`
- To call the library we write `std::vector<double>`
- We can be lazy and write `using namespace std;`

Namespaces

- When you are writing very large programmes (possibly involving other peoples code) you might accidentally use the same name for a class, function or variable used elsewhere
- If you are luck this won't compile, or crash. If you are unlucky you will have a weird bug that will be very difficult to find
- To prevent this, C++ invented a new scope called **namespaces**
- By default all the standard library classes and functions are in namespace `std`
- To call the library we write `std::vector<double>`
- We can be lazy and write `using namespace std;`

Namespaces

- When you are writing very large programmes (possibly involving other peoples code) you might accidentally use the same name for a class, function or variable used elsewhere
- If you are luck this won't compile, or crash. If you are unlucky you will have a weird bug that will be very difficult to find
- To prevent this, C++ invented a new scope called **namespaces**
- By default all the standard library classes and functions are in namespace `std`
- To call the library we write `std::vector<double>`
- We can be lazy and write `using namespace std;`

Namespaces

- When you are writing very large programmes (possibly involving other peoples code) you might accidentally use the same name for a class, function or variable used elsewhere
- If you are luck this won't compile, or crash. If you are unlucky you will have a weird bug that will be very difficult to find
- To prevent this, C++ invented a new scope called **namespaces**
- By default all the standard library classes and functions are in namespace `std`
- To call the library we write `std::vector<double>`
- We can be lazy and write `using namespace std;`

Print

- Rather than pesky `printf` statements C++ allows us to use the opeartor <<

- When you get used to it, you will love if

```
#include <iostream>      // head file the defines library
using namespace std;

void main() {
  int i = 5;
  double x = 3.3;

  cout << "hello there" << i << ' ' << x << endl;
}
```

Outline

1. C with Classes
2. **New**
3. Overloading
4. Templates

IN

CONGRESS,

JULY 4, 1776.

A

DECLARATION

BY THE

REPRESENTATIVES

OF THE

UNITED STATES OF AMERICA,

In GENERAL CONGRESS assembled.

WHEN in the course of human events, it becomes necessary for one people to dissolve the political bands which have connected them with another, and to assume among the powers of the earth, the separate and equal station to which the laws of nature and of nature's God entitle them, a decent respect to the opinions of mankind requires that they should declare the causes which impel them to the separation.

We hold these truths to be self-evident, that all men are created equal, that they are endowed by their Creator with certain unalienable rights, that among these are life, Liberty, and the pursuit of happiness. That to secure these rights, governments are instituted among men, deriving their just powers from the consent of the governed, that whenever any form of government becomes destructive of these ends, it is the right of the people to alter or to abolish it, and to institute new government, laying its foundation on such principles, and organizing its powers in such form, as to them shall seem most likely to effect their safety and happiness. Prudence, indeed, will dictate that governments long established should not be changed for light and transient causes; and accordingly all experience hath shewn, that mankind are more disposed to suffer, while evils are sufferable, than to right themselves by abolishing the forms to which they are accustomed. But when a long train of abuses and usurpations, pursuing invariably the same object, evinces a design to reduce them under absolute despotism, it is their right, it is their duty, to throw off such government, and to provide new guards for their future security. Such has been the patient sufferance of these Colonies; and such is now the necessity which constrains them to alter their former systems of government. The history of the present King of Great-Britain is a history of repeated injuries and usurpations, all having in direct object the establishment of an absolute tyranny over these states. To prove this, let facts be submitted to a candid world.

He has refused his assent to laws, the most wholesome and necessary for the public good.

He has forbidden his Governors to pass laws of immediate and pressing importance, unless suspended in their operation till his assent should be obtained; and when so suspended, he has utterly neglected to attend them.

He has refused to pass other laws for the accommodation of large districts of people, unless those people would relinquish the right of representation in the legislature, a right inestimable to them, and formidable to Tyrants only.

He has called together legislative bodies at places unusual, uncomfortable, and distant from the depository of their public records, for the sole purpose of fatiguing them into compliance with his measures.

He has dissolved representative houses repeatedly, for opposing with manly firmness his invasions on the rights of the people.

He has refused for a long time, after such dissolutions, to cause others to be elected; whereby the legislative powers, incapable of annihilation, have returned to the people at large for their exercise; the state remaining in the mean time exposed to all the dangers of invasion from without, and convulsions within.

He has endeavored to prevent the population of these States; for that purpose obstructing the laws for naturalization of foreigners; refusing to pass others to encourage their migrations hither, and raising the conditions of new appropriations of lands.

He has obstructed the administration of justice, by refusing his assent to laws for establishing judiciary powers.

He has made Judges dependent on his will alone, for the tenure of their offices, and the amount and payment of their salaries.

He has erected a multitude of new offices, and sent hither swarms of officers to harass our people, and eat out their substance.

He has kept among us, in times of peace, standing armies, without the consent of our Legislatures.

He has affected to render the military independent of and superior to the civil power.

He has combined with others to subject us to a jurisdiction foreign to our constitution, and unacknowledged by our laws; giving his assent to their acts of pretended legislation:

For quartering large bodies of armed troops among us:

For protecting them, by a mock trial, from punishment for any murders which they should commit on the inhabitants of these states:

For cutting off our trade with all parts of the world:

For imposing taxes on us without our consent:

For depriving us, in many cases, of the benefits of trial by jury:

For transporting us beyond seas to be tried for pretended offences:

For abolishing the free system of English laws in a neighboring province, establishing therein an arbitrary government, and enlarging its boundaries, so as to render it at once an example and fit instrument for introducing the same absolute rule into these Colonies:

For taking away our Charters, abolishing our most valuable laws, and altering fundamentally the forms of our governments:

For suspending our own Legislatures, and declaring themselves invested with power to legislate for us in all cases whatsoever.

He has abdicated government here, by declaring us out of his protection and waging war against us.

He has plundered our seas, ravaged our coasts, burnt our towns, and destroyed the lives of our people.

He is at this time, transporting large armies of foreign mercenaries to compleat the works of death, desolation and tyranny, already begun with circumstances of cruelty and perfidy scarcely paralleled in the most barbarous ages, and totally unworthy the head of a civilized nation.

He has constrained our fellow citizens taken captive on the high seas to bear arms against their country, to become the executioners of their friends and brethren, or to fall themselves by their hands.

He has excited domestic insurrections amongst us, and has endeavored to bring on the inhabitants of our frontiers, the merciless Indian Savages, whose known rule of warfare, is an undistinguished destruction of all ages, sexes, and conditions.

In every stage of these oppressions we have petitioned for redress, in the most humble terms: Our repeated petitions have been answered only by repeated injury. A Prince, whose character is thus marked by every act which may define a Tyrant, is unfit to be the Ruler of a free People.

Nor have we been wanting in attention to our British brethren. We have warned them from time to time of attempts by their legislature to extend an unwarrantable jurisdiction over us. We have reminded them of the circumstances of our emigration and settlement here. We have appealed to their native justice and magnanimity, and we have conjured them by the ties of our common kindred to disavow these usurpations, which would inevitably interrupt our connections and correspondence. They too have been deaf to the voice of justice and of consanguinity. We must, therefore, acquiesce in the necessity which denounces our separation, and hold them, as we hold the rest of mankind, enemies in war; in peace, friends.

We, therefore, the Representatives of the UNITED STATES OF AMERICA, in GENERAL CONGRESS assembled, appealing to the Supreme Judge of the world for the rectitude of our intentions, do in the name and by the authority of the good People of these Colonies, solemnly publish and declare, That these United Colonies are, and of right ought to be, FREE AND INDEPENDENT STATES; that they are absolved from all allegiance to the British Crown, and that all political connection between them and the State of Great-Britain, is, and ought to be totally dissolved; and that as FREE AND INDEPENDENT STATES, they have full power to levy war, conclude peace, contract alliances, establish commerce, and to do all other acts and things which INDEPENDENT STATES may of right do. And for the support of this Declaration, with a firm reliance on the protection of Divine Providence, we mutually pledge to each other our lives, our fortunes, and our sacred honor.

Signed by ORDER *and in behalf of the* CONGRESS,

JOHN HANCOCK, PRESIDENT.

ATTEST,

CHARLES THOMPSON, Secretary.

Pointers

- In C and C++ we can access an object through its memory address

```
int a = 5;      // creates an object a with value 5
int* b = &a;    // b is the memory address of object a
*b = 6          // *b is now a pseudonym for a
```

- b is called a pointer

- The *dereferencing* operator * turns the pointer back into the object

Pointers

- In C and C++ we can access an object through its memory address

```
int a = 5;      // creates an object a with value 5
int* b = &a;    // b is the memory address of object a
*b = 6          // *b is now a pseudonym for a
```

- b is called a pointer

- The *dereferencing* operator * turns the pointer back into the object

Pointers

- In C and C++ we can access an object through its memory address

```
int a = 5;      // creates an object a with value 5
int* b = &a;    // b is the memory address of object a
*b = 6          // *b is now a pseudonym for a
```

- b is called a pointer

- The *dereferencing* operator * turns the pointer back into the object

Pointers

- In C and C++ we can access an object through its memory address

```
int a = 5;      // creates an object a with value 5
int* b = &a;    // b is the memory address of object a
*b = 6          // *b is now a pseudonym for a
```

- b is called a pointer

- The *dereferencing* operator * turns the pointer back into the object

Pointers

- In C and C++ we can access an object through its memory address

```
int a = 5;      // creates an object a with value 5
int* b = &a;    // b is the memory address of object a
*b = 6          // *b is now a pseudonym for a
```

- b is called a pointer

- The *dereferencing* operator * turns the pointer back into the object

New Object

- The operator **new** will create an object and return a reference

```
Widget w(arg);                      // w is an instance of class Widget
Widget* wpt = new Widget(args); // pointer to instance of class Widget
```

- To call a member function of wp use either

```
(*wpt).func();   // dereference object and call member function
wpt->func();     // easy to type
```

New Object

- The operator **new** will create an object and return a reference

```
Widget w(arg);                    // w is an instance of class Widget
Widget* wpt = new Widget(args); // pointer to instance of class Widget
```

- To call a member function of wp use either

```
(*wpt).func();  // dereference object and call member function
wpt->func();    // easy to type
```

Inheritence

- C++ allows classes to inherit from other classes
- `Square` and `Circle` might inherit from `Shape`
- If `Shape` has a (virtual) member function `area` then `Square` and `Circle` might redefine this

```
class Square: public Shape {
  private:
    double l;

  public:
    Square(double len) {l=len;}   // constructor
    double area() {return l*l;}   // define area
}
```

Inheritence

- C++ allows classes to inherit from other classes
- `Square` and `Circle` might inherit from `Shape`
- If `Shape` has a (virtual) member function `area` then `Square` and `Circle` might redefine this

```
class Square: public Shape {
  private:
    double l;

  public:
    Square(double len) {l=len;}   // constructor
    double area() {return l*l;}   // define area
}
```

Inheritence

- C++ allows classes to inherit from other classes
- `Square` and `Circle` might inherit from `Shape`
- If `Shape` has a (virtual) member function `area` then `Square` and `Circle` might redefine this

```
class Square: public Shape {
  private:
    double l;

  public:
    Square(double len) {l=len;}   // constructor
    double area() {return l*l;}   // define area
}
```

Polymorphism

- Polymorphism is a way of using inheritance where we instantiate a parent pointer with a child class

```
Shape* shape = new Square(2.5);

cout << shape->area() << endl;
```

- This provides a clean way of choosing a behaviour depending on the object type

- It is used in *iterator*s which we will come to later in the course

Polymorphism

- Polymorphism is a way of using inheritance where we instantiate a parent pointer with a child class

```
Shape* shape = new Square(2.5);

cout << shape->area() << endl;
```

- This provides a clean way of choosing a behaviour depending on the object type

- It is used in *iterator*s which we will come to later in the course

Polymorphism

- Polymorphism is a way of using inheritance where we instantiate a parent pointer with a child class

```
Shape* shape = new Square(2.5);

cout << shape->area() << endl;
```

- This provides a clean way of choosing a behaviour depending on the object type

- It is used in *iterator*s which we will come to later in the course

Polymorphism

- Polymorphism is a way of using inheritance where we instantiate a parent pointer with a child class

```
Shape* shape = new Square(2.5);

cout << shape->area() << endl;
```

- This provides a clean way of choosing a behaviour depending on the object type

- It is used in *iterator*s which we will come to later in the course

Arrays

- C++ also uses **new** to return arrays (in place of malloc)

```
int* pt = new int[20];
```

  creates a pointer to memory location where we can store 20 integers

- We can dereference the i^{th} element using `pt[i]`

- We can free this up with

```
delete[] pt;
```

Arrays

- C++ also uses **new** to return arrays (in place of malloc)

```
int* pt = new int[20];
```

  creates a pointer to memory location where we can store 20 integers

- We can dereference the i^{th} element using `pt[i]`

- We can free this up with

```
delete[] pt;
```

Arrays

- C++ also uses **new** to return arrays (in place of malloc)

```
int* pt = new int[20];
```

  creates a pointer to memory location where we can store 20 integers

- We can dereference the i^{th} element using `pt[i]`

- We can free this up with

```
delete[] pt;
```

References

- C and C++ also provides references

```
int a = 5;          // create a memory location called a
int& b = a;         // b is a pseudonym for a
b = 6               // both b and a are now 6
```

- References are like dereferenced pointers

- There are many uses of references, one is so we can make functions change their value

```
void f(int x) {x += 6;}   // define function f

void g(int& x) {x += 2;}  // define function g

int a = 5;

f(a);                      // does nothing a=5
g(a);                      // now a=7
```

References

- C and C++ also provides references

```
int a = 5;          // create a memory location called a
int& b = a;         // b is a pseudonym for a
b = 6               // both b and a are now 6
```

- References are like dereferenced pointers

- There are many uses of references, one is so we can make functions change their value

```
void f(int x) {x += 6;}   // define function f

void g(int& x) {x += 2;}  // define function g

int a = 5;

f(a);                      // does nothing a=5
g(a);                      // now a=7
```

References

- C and C++ also provides references

```
int a = 5;          // create a memory location called a
int& b = a;         // b is a pseudonym for a
b = 6               // both b and a are now 6
```

- References are like dereferenced pointers

- There are many uses of references, one is so we can make functions change their value

```
void f(int x) {x += 6;}   // define function f

void g(int& x) {x += 2;}  // define function g

int a = 5;

f(a);                      // does nothing a=5
g(a);                      // now a=7
```

References

- C and C++ also provides references

```
int a = 5;          // create a memory location called a
int& b = a;         // b is a pseudonym for a
b = 6               // both b and a are now 6
```

- References are like dereferenced pointers

- There are many uses of references, one is so we can make functions change their value

```
void f(int x) {x += 6;}   // define function f

void g(int& x) {x += 2;}  // define function g

int a = 5;

f(a);                      // does nothing a=5
g(a);                      // now a=7
```

References

- C and C++ also provides references

```
int a = 5;          // create a memory location called a
int& b = a;         // b is a pseudonym for a
b = 6               // both b and a are now 6
```

- References are like dereferenced pointers

- There are many uses of references, one is so we can make functions change their value

```
void f(int x) {x += 6;}    // define function f

void g(int& x) {x += 2;}   // define function g

int a = 5;

f(a);                      // does nothing a=5
g(a);                      // now a=7
```

References

- C and C++ also provides references

```
int a = 5;          // create a memory location called a
int& b = a;         // b is a pseudonym for a
b = 6               // both b and a are now 6
```

- References are like dereferenced pointers

- There are many uses of references, one is so we can make functions change their value

```
void f(int x) {x += 6;}   // define function f

void g(int& x) {x += 2;}  // define function g

int a = 5;

f(a);                      // does nothing a=5
g(a);                      // now a=7
```

References

- C and C++ also provides references

```
int a = 5;          // create a memory location called a
int& b = a;         // b is a pseudonym for a
b = 6               // both b and a are now 6
```

- References are like dereferenced pointers

- There are many uses of references, one is so we can make functions change their value

```
void f(int x) {x += 6;}   // define function f

void g(int& x) {x += 2;}  // define function g

int a = 5;

f(a);                      // does nothing a=5
g(a);                      // now a=7
```

References

- C and C++ also provides references

```
int a = 5;          // create a memory location called a
int& b = a;         // b is a pseudonym for a
b = 6               // both b and a are now 6
```

- References are like dereferenced pointers

- There are many uses of references, one is so we can make functions change their value

```
void f(int x) {x += 6;}   // define function f

void g(int& x) {x += 2;}  // define function g

int a = 5;

f(a);                     // does nothing a=5
g(a);                     // now a=7
```

References

- C and C++ also provides references

```
int a = 5;          // create a memory location called a
int& b = a;         // b is a pseudonym for a
b = 6               // both b and a are now 6
```

- References are like dereferenced pointers

- There are many uses of references, one is so we can make functions change their value

```
void f(int x) {x += 6;}   // define function f

void g(int& x) {x += 2;}  // define function g

int a = 5;

f(a);                      // does nothing a=5
g(a);                      // now a=7
```

Saving Copying

- When we declare a function `f(Widget w)` then widget `w` is copied to the function (this is known as passed by value)
- If widget is big, even if we don't want to change it we might not want to copy it

```
void f(const Widget& w);
void g(Widget w);
```

- In both cases `w` is a Widget, but function `f` avoids copying its input

Saving Copying

- When we declare a function `f(Widget w)` then widget `w` is copied to the function (this is known as passed by value)
- If widget is big, even if we don't want to change it we might not want to copy it

```
void f(const Widget& w);
void g(Widget w);
```

- In both cases `w` is a Widget, but function `f` avoids copying its input

Saving Copying

- When we declare a function `f(Widget w)` then widget `w` is copied to the function (this is known as passed by value)
- If widget is big, even if we don't want to change it we might not want to copy it

```
void f(const Widget& w);
void g(Widget w);
```

- In both cases `w` is a Widget, but function `f` avoids copying its input

Saving Copying

- When we declare a function `f(Widget w)` then widget `w` is copied to the function (this is known as passed by value)
- If widget is big, even if we don't want to change it we might not want to copy it

```
void f(const Widget& w);
void g(Widget w);
```

- In both cases `w` is a Widget, but function `f` avoids copying its input

Outline

1. C with Classes
2. New
3. **Overloading**
4. Templates

IN

CONGRESS,

JULY 4, 1776.

A

DECLARATION

BY THE

REPRESENTATIVES

OF THE

UNITED STATES OF AMERICA,

In GENERAL CONGRESS assembled.

WHEN in the course of human events, it becomes necessary for one people to dissolve the political bands which have connected them with another, and to assume among the powers of the earth, the separate and equal station to which the laws of nature and of nature's God entitle them, a decent respect to the opinions of mankind requires that they should declare the causes which impel them to the separation.

We hold these truths to be self-evident, that all men are created equal, that they are endowed by their Creator with certain unalienable rights, that among these are life, Liberty, and the pursuit of happiness. That to secure these rights, governments are instituted among men, deriving their just powers from the consent of the governed, that whenever any form of government becomes destructive of these ends, it is the right of the people to alter or to abolish it, and to institute new government, laying its foundation on such principles, and organizing its powers in such form, as to them shall seem most likely to effect their safety and happiness. Prudence, indeed, will dictate that governments long established should not be changed for light and transient causes; and accordingly all experience hath shewn, that mankind are more disposed to suffer, while evils are sufferable, than to right themselves by abolishing the forms to which they are accustomed. But when a long train of abuses and usurpations, pursuing invariably the same object, evinces a design to reduce them under absolute despotism, it is their right, it is their duty, to throw off such government, and to provide new guards for their future security. Such has been the patient sufferance of these Colonies; and such is now the necessity which constrains them to alter their former systems of government. The history of the present King of Great-Britain is a history of repeated injuries and usurpations, all having in direct object the establishment of an absolute tyranny over these states. To prove this, let facts be submitted to a candid world.

He has refused his assent to laws, the most wholesome and necessary for the public good.

He has forbidden his Governors to pass laws of immediate and pressing importance, unless suspended in their operation till his assent should be obtained; and when so suspended, he has utterly neglected to attend them.

He has refused to pass other laws for the accommodation of large districts of people, unless those people would relinquish the right of representation in the legislature, a right inestimable to them, and formidable to tyrants only.

He has called together legislative bodies at places unusual, uncomfortable, and distant from the depository of their public records, for the sole purpose of fatiguing them into compliance with his measures.

He has dissolved representative houses repeatedly, for opposing with manly firmness his invasions on the rights of the people.

He has refused for a long time, after such dissolutions, to cause others to be elected; whereby the legislative powers, incapable of annihilation, have returned to the people at large for their exercise; the state remaining in the mean time exposed to all the dangers of invasion from without, and convulsions within.

He has endeavored to prevent the population of these states; for that purpose obstructing the laws for naturalization of foreigners; refusing to pass others to encourage their migrations hither, and raising the conditions of new appropriations of lands.

He has obstructed the administration of justice, by refusing his assent to laws for establishing judiciary powers.

He has made Judges dependent on his will alone, for the tenure of their offices, and the amount and payment of their salaries.

He has erected a multitude of new offices, and sent hither swarms of officers to harass our people, and eat out their substance.

He has kept among us, in times of peace, standing armies, without the consent of our Legislatures.

He has affected to render the military independent of and superior to the civil power.

He has combined with others to subject us to a jurisdiction foreign to our constitution, and unacknowledged by our laws; giving his assent to their acts of pretended legislation:

For quartering large bodies of armed troops among us:

For protecting them, by a mock trial, from punishment for any murders which they should commit on the inhabitants of these states:

For cutting off our trade with all parts of the world:

For imposing taxes on us without our consent:

For depriving us, in many cases, of the benefits of trial by jury:

For transporting us beyond seas to be tried for pretended offences:

For abolishing the free system of English laws in a neighboring province, establishing therein an arbitrary government, and enlarging its boundaries, so as to render it at once an example and fit instrument for introducing the same absolute rule into these Colonies:

For taking away our Charters, abolishing our most valuable laws, and altering fundamentally the forms of our governments:

For suspending our own Legislatures, and declaring themselves invested with power to legislate for us in all cases whatsoever.

He has abdicated government here, by declaring us out of his protection and waging war against us.

He has plundered our seas, ravaged our coasts, burnt our towns, and destroyed the lives of our people.

He is at this time, transporting large armies of foreign mercenaries to compleat the works of death, desolation and tyranny, already begun with circumstances of cruelty and perfidy scarcely paralleled in the most barbarous ages, and totally unworthy the head of a civilized nation.

He has constrained our fellow citizens taken captive on the high seas to bear arms against their country, to become the executioners of their friends and brethren, or to fall themselves by their hands.

He has excited domestic insurrections amongst us, and has endeavored to bring on the inhabitants of our frontiers, the merciless Indian Savages, whose known rule of warfare, is an undistinguished destruction of all ages, sexes, and conditions.

In every stage of these oppressions we have petitioned for redress, in the most humble terms: Our repeated petitions have been answered only by repeated injury. A Prince, whose character is thus marked by every act which may define a Tyrant, is unfit to be the Ruler of a free People.

Nor have we been wanting in attention to our British brethren. We have warned them from time to time of attempts by their legislature to extend an unwarrantable jurisdiction over us. We have reminded them of the circumstances of our emigration and settlement here. We have appealed to their native justice and magnanimity, and we have conjured them by the ties of our common kindred to disavow these usurpations, which would inevitably interrupt our connections and correspondence. They too have been deaf to the voice of justice and of consanguinity. We must, therefore, acquiesce in the necessity which denounces our separation, and hold them, as we hold the rest of mankind, enemies in war; in peace, friends.

We, therefore, the Representatives of the UNITED STATES OF AMERICA, in GENERAL CONGRESS assembled, appealing to the Supreme Judge of the world for the rectitude of our intentions, do in the name and by the authority of the good People of these Colonies, solemnly publish and declare, That these United Colonies are, and of right ought to be, FREE AND INDEPENDENT STATES; that they are absolved from all allegiance to the British Crown, and that all political connection between them and the State of Great-Britain, is, and ought to be totally dissolved; and that as FREE AND INDEPENDENT STATES, they have full power to levy war, conclude peace, contract alliances, establish commerce, and to do all other acts and things which INDEPENDENT STATES may of right do. And for the Support of this Declaration, with a firm reliance on the protection of Divine Providence, we mutually pledge to each other our lives, our fortunes, and our sacred honor.

Signed by ORDER *and in behalf of the* CONGRESS,

JOHN HANCOCK, PRESIDENT.

ATTEST,

CHARLES THOMPSON, Secretary.

Overloading

- C and C++ allow you to define different functions with the same name but different arguments

```
void func(int a);      // called if argument is an int
void func(double a);   // called if argument is a double
```

- Needs to be used sensibly, but provides flexibility

Example

- In the second order statistics class we could define a member function

```
void add(const Sos& rhs);
```

- With an implementation

```
void Sos::add(const Sos& rhs)
{
  double total = n + rhs.n;
  double diff = rhs.mu-mu;
  mu += rhs.n*diff/total;
  Q += rhs.Q + n*rhs.n*diff*diff/total;
  n = total;

  return rhs;
}
```

Overloading Continued

- This allows us to add second order statistics

```
Sos total;
for(int i=0; i<10; ++i) {
  Sos local;
  for(int j=0; j<100; ++j) {
    // compute X
    cout << local.mean() << ',' << local.error() << endl;
    local.add()
  }
  total.add(local)
  cout << total.mean() << ',' << total.error() << endl;
}
```

Opeartor Overloading

- C++ like python allows us to overload operators
- Rather than using add I might prefer to use

```
class Sos {
  ...
  double operator+=(double x) { add(x); return(x); }
}
```

- Then we can write

```
Sos sos;
sos += X;
```

Overloading <<

- To print an object of type `Sos` we define

```
ostream& operator<<(ostream& out, const Sos& d)
{
  out << d.mean() << " " << d.error();
  return(out);
}
```

- We can then print

```
Sos sos;
...

cout << sos << endl;
```

Outline

1. C with Classes
2. New
3. Overloading
4. **Templates**

IN
CONGRESS,
JULY 4, 1776.
A
DECLARATION
BY THE
REPRESENTATIVES
OF THE
UNITED STATES OF AMERICA,
In GENERAL CONGRESS assembled.

WHEN in the course of human events, it becomes necessary for one people to dissolve the political bands which have connected them with another, and to assume among the powers of the earth, the separate and equal station to which the laws of nature and of nature's God entitle them, a decent respect to the opinions of mankind requires that they should declare the causes which impel them to the separation.

We hold these truths to be self-evident, that all men are created equal, that they are endowed by their Creator with certain unalienable rights, that among these are life, Liberty, and the pursuit of happiness. That to secure these rights, governments are instituted among men, deriving their just powers from the consent of the governed, that whenever any form of government becomes destructive of these ends, it is the right of the people to alter or to abolish it, and to institute new government, laying its foundation on such principles, and organizing its powers in such form, as to them shall seem most likely to effect their safety and happiness. Prudence, indeed, will dictate that governments long established should not be changed for light and transient causes; and accordingly all experience hath shewn, that mankind are more disposed to suffer, while evils are sufferable, than to right themselves by abolishing the forms to which they are accustomed. But when a long train of abuses and usurpations, pursuing invariably the same object, evinces a design to reduce them under absolute despotism, it is their right, it is their duty, to throw off such government, and to provide new guards for their future security. Such has been the patient sufferance of these Colonies; and such is now the necessity which constrains them to alter their former systems of government. The history of the present King of Great-Britain is a history of repeated injuries and usurpations, all having in direct object the establishment of an absolute tyranny over these states. To prove this, let facts be submitted to a candid world.

He has refused his assent to laws, the most wholesome and necessary for the public good.

He has forbidden his Governors to pass laws of immediate and pressing importance, unless suspended in their operation till his assent should be obtained; and when so suspended, he has utterly neglected to attend them.

He has refused to pass other laws for the accommodation of large districts of people, unless those people would relinquish the right of representation in the legislature, a right inestimable to them, and formidable to tyrants only.

He has called together legislative bodies at places unusual, uncomfortable, and distant from the depository of their public records, for the sole purpose of fatiguing them into compliance with his measures.

He has dissolved representative houses repeatedly, for opposing with manly firmness his invasions on the rights of the people.

He has refused for a long time, after such dissolutions, to cause others to be elected; whereby the legislative powers, incapable of annihilation, have returned to the people at large for their exercise; the state remaining in the mean time exposed to all the dangers of invasion from without, and convulsions within.

He has endeavored to prevent the population of these states; for that purpose obstructing the laws for naturalization of foreigners; refusing to pass others to encourage their migrations hither, and raising the conditions of new appropriations of lands.

He has obstructed the administration of justice, by refusing his assent to laws for establishing judiciary powers.

He has made Judges dependent on his will alone, for the tenure of their offices, and the amount and payment of their salaries.

He has erected a multitude of new offices, and sent hither swarms of officers to harass our people, and eat out their substance.

He has kept among us, in times of peace, standing armies, without the consent of our Legislatures.

He has affected to render the military independent of and superior to the civil power.

He has combined with others to subject us to a jurisdiction foreign to our constitution, and unacknowledged by our laws; giving his assent to their acts of pretended legislation:

For quartering large bodies of armed troops among us:

For protecting them, by a mock trial, from punishment for any murders which they should commit on the inhabitants of these states:

For cutting off our trade with all parts of the world:

For imposing taxes on us without our consent:

For depriving us, in many cases, of the benefits of trial by jury:

For transporting us beyond seas to be tried for pretended offenses:

For abolishing the free system of English laws in a neighboring province, establishing therein an arbitrary government, and enlarging its boundaries, so as to render it at once an example and fit instrument for introducing the same absolute rule into these Colonies:

For taking away our Charters, abolishing our most valuable laws, and altering fundamentally the forms of our governments:

For suspending our own Legislatures, and declaring themselves invested with power to legislate for us in all cases whatsoever.

He has abdicated government here, by declaring us out of his protection and waging war against us.

He has plundered our seas, ravaged our coasts, burnt our towns, and destroyed the lives of our people.

He is at this time, transporting large armies of foreign mercenaries to compleat the works of death, desolation and tyranny, already begun with circumstances of cruelty and perfidy scarcely paralleled in the most barbarous ages, and totally unworthy the head of a civilized nation.

He has constrained our fellow citizens taken captive on the high seas to bear arms against their country, to become the executioners of their friends and brethren, or to fall themselves by their hands.

He has excited domestic insurrections amongst us, and has endeavored to bring on the inhabitants of our frontiers, the merciless Indian Savages, whose known rule of warfare, is an undistinguished destruction of all ages, sexes, and conditions.

In every stage of these oppressions we have petitioned for redress, in the most humble terms: Our repeated petitions have been answered only by repeated injury. A Prince, whose character is thus marked by every act which may define a Tyrant, is unfit to be the Ruler of a free People.

Nor have we been wanting in attentions to our British brethren. We have warned them from time to time of attempts by their legislature to extend an unwarrantable jurisdiction over us. We have reminded them of the circumstances of our emigration and settlement here. We have appealed to their native justice and magnanimity, and we have conjured them by the ties of our common kindred to disavow these usurpations, which would inevitably interrupt our connections and correspondence. They too have been deaf to the voice of justice and of consanguinity. We must, therefore, acquiesce in the necessity which denounces our separation, and hold them, as we hold the rest of mankind, enemies in war; in peace, friends.

We, therefore, the Representatives of the UNITED STATES OF AMERICA, in GENERAL CONGRESS assembled, appealing to the Supreme Judge of the world for the rectitude of our intentions, do in the name and by the authority of the good People of these Colonies, solemnly publish and declare, That these United Colonies are, and of right ought to be, FREE AND INDEPENDENT STATES; that they are absolved from all allegiance to the British Crown, and that all political connection between them and the State of Great-Britain, is, and ought to be totally dissolved; and that as FREE AND INDEPENDENT STATES, they have full power to levy war, conclude peace, contract alliances, establish commerce, and to do all other acts and things which INDEPENDENT STATES may of right do. And for the Support of this Declaration, with a firm reliance on the protection of Divine Providence, we mutually pledge to each other our lives, our fortunes, and our sacred honor.

Signed by ORDER *and in behalf of the* CONGRESS,
JOHN HANCOCK, PRESIDENT.
ATTEST,
CHARLES THOMPSON, Secretary.

Templates

- Many algorithms and data structures can be applied to a wide range of types

```
vector<double> double_vec;   // resizable array of doubles
vector<int>    int_vec;      // resizable array of int
map<string, int>  mymap      // map with string keys and int valu
```

- C++ allows us to define a template class

```
template <typename T>
class myclass {
  private T data;
}
```

Templates

- Many algorithms and data structures can be applied to a wide range of types

```
vector<double> double_vec;   // resizable array of doubles
vector<int>    int_vec;      // resizable array of int
map<string, int>  mymap      // map with string keys and int valu
```

- C++ allows us to define a template class

```
template <typename T>
class myclass {
  private T data;
}
```

Templates

- Many algorithms and data structures can be applied to a wide range of types

```
vector<double> double_vec;   // resizable array of doubles
vector<int>    int_vec;      // resizable array of int
map<string, int>  mymap      // map with string keys and int valu
```

- C++ allows us to define a template class

```
template <typename T>
class myclass {
  private T data;
}
```

Templates

- Many algorithms and data structures can be applied to a wide range of types

```
vector<double> double_vec;   // resizable array of doubles
vector<int>    int_vec;      // resizable array of int
map<string, int>  mymap      // map with string keys and int valu
```

- C++ allows us to define a template class

```
template <typename T>
class myclass {
  private T data;
}
```

Templates

- Many algorithms and data structures can be applied to a wide range of types

```
vector<double> double_vec;   // resizable array of doubles
vector<int>    int_vec;      // resizable array of int
map<string, int>  mymap      // map with string keys and int valu
```

- C++ allows us to define a template class

```
template <typename T>
class myclass {
  private T data;
}
```

Templates

- Templates work very simply
- They provide a template for same type (e.g. `T`)
- When you ask for an instance of that object

```
myclass<int> instance;
```

  the C++ compiler takes your template and substitutes the `T` with **`int`**

- This is both simple and powerful

Templates

- Templates work very simply
- They provide a template for same type (e.g. `T`)
- When you ask for an instance of that object

```
myclass<int> instance;
```

  the C++ compiler takes your template and substitutes the `T` with **`int`**

- This is both simple and powerful

Templates

- Templates work very simply
- They provide a template for same type (e.g. `T`)
- When you ask for an instance of that object

```
myclass<int> instance;
```

  the C++ compiler takes your template and substitutes the `T` with **`int`**

- This is both simple and powerful

Templates

- Templates work very simply
- They provide a template for same type (e.g. `T`)
- When you ask for an instance of that object

```
myclass<int> instance;
```

  the C++ compiler takes your template and substitutes the `T` with **`int`**

- This is both simple and powerful

Templates

- Templates work very simply
- They provide a template for same type (e.g. `T`)
- When you ask for an instance of that object

```
myclass<int> instance;
```

  the C++ compiler takes your template and substitutes the `T` with **`int`**

- This is both simple and powerful

Template Functions

- As well as classes I can create template functions

```
template <typename T>
T accumulate(const vector<T>& vec) {
  T sum = 0;
  for(int i=0; i<vec.size(); ++i) {
    sum += vec[i];
  }
  return sum
}
```

- This will work with `vector<int>`, `vector<double>`

Template Functions

- As well as classes I can create template functions

```
template <typename T>
T accumulate(const vector<T>& vec) {
  T sum = 0;
  for(int i=0; i<vec.size(); ++i) {
    sum += vec[i];
  }
  return sum
}
```

- This will work with `vector<int>`, `vector<double>`

Summary

- C++ is a rich language
- You should learn some C++ in low-level programming
- There are a lot of resources
- I'm afraid you will only get good at it by writing programs
- The lab session are to help you learn C++

Summary

- C++ is a rich language
- You should learn some C++ in low-level programming
- There are a lot of resources
- I'm afraid you will only get good at it by writing programs
- The lab session are to help you learn C++

Summary

- C++ is a rich language
- You should learn some C++ in low-level programming
- There are a lot of resources
- I'm afraid you will only get good at it by writing programs
- The lab session are to help you learn C++

Summary

- C++ is a rich language
- You should learn some C++ in low-level programming
- There are a lot of resources
- I'm afraid you will only get good at it by writing programs
- The lab session are to help you learn C++

Summary

- C++ is a rich language
- You should learn some C++ in low-level programming
- There are a lot of resources
- I'm afraid you will only get good at it by writing programs
- The lab session are to help you learn C++