Intermediate/Advanced Computer Programming

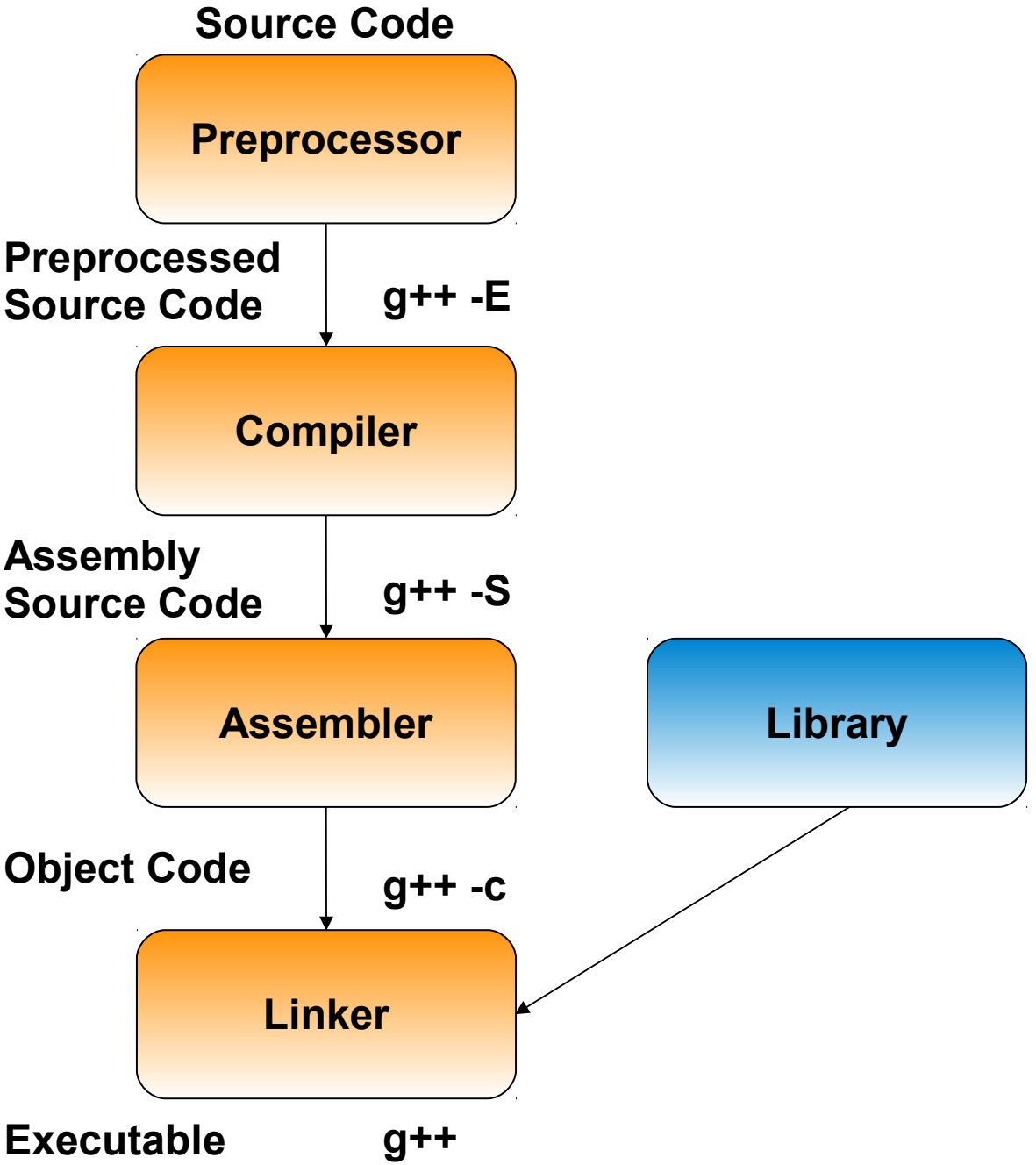
Review

- What command is used to create an executable from C++ source code?
- How can a file be used as input/output for a program?
- How are variables/arrays defined in C++?
- List a reasonable set of steps to creating a program
- What is the syntax for comments in C++?
- Is correct program operation is the only consideration when writing a program?

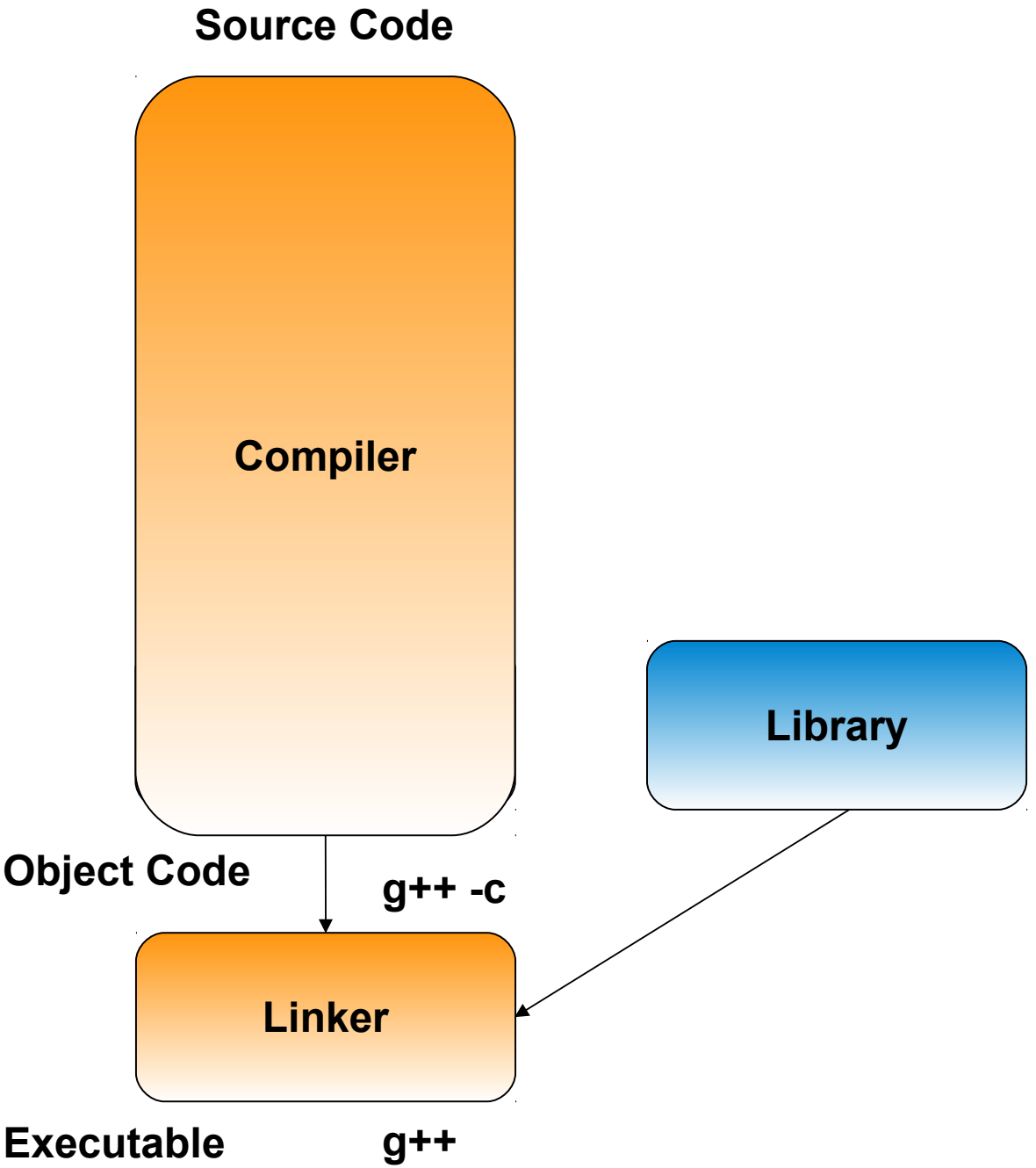
Code Evaluation

- Correctness
- Quality of Code
  - Readability
  - Conveying program semantics to the compiler
  - Scalability
  - Efficiency

Building an Executable



Building an Executable



Loops (Iteration Structures)

- Sequence of statements that can be executed more than once in succession
- Allows a statement to be repeated a certain number of times or while a condition is fulfilled.

while Loop

while (condition)

body-statement;

```
int x = 5;
while (x>0) {
    cout << x << endl;
    x = x - 1;
}
```

do-while Loop

do

statement;

while (condition);

```
do
     cout << "enter a number(-1 to exit):";
     cin >> number;
while (number != -1);
```

for Loops

for (initializer ; condition ; iteration statement)

body-statement;

- Write a for-loop that will iterate x number of times?
- How can we write an equivalent while loop?

Loops

for (initialization ; condition ; iteration statement)

body-statement;

initialization;

while (*condition*) {

body-statement;

iteration statement;

}

Exit and Continuation (Loops)

break

exits out of the innermost enclosing loop

continue

Begins a new iteration of the innermost enclosing loop

Exit and Continuation (Loops)

```
for (int i=0; i<20; ++i) {
    if (!(i%4)) continue;
    cout << i << endl;
}
```

Do we need the parentheses around i%4?

Exit and Continuation (Loops)

```
for (int i=0; i<20; ++i) {
    if (arr[i] == 3) {
        cout << "3 found" << endl;
        break;
    }
}
```

What would happen if we accidentally 'wrote arr[i] = 3'?

OOP

OOP

Attempts to solve problems when writing code:

- More structure
- More reusable
- More inline with how how humans think
- More ways to protect data from being improperly accessed
- Less worry about coding intricacies

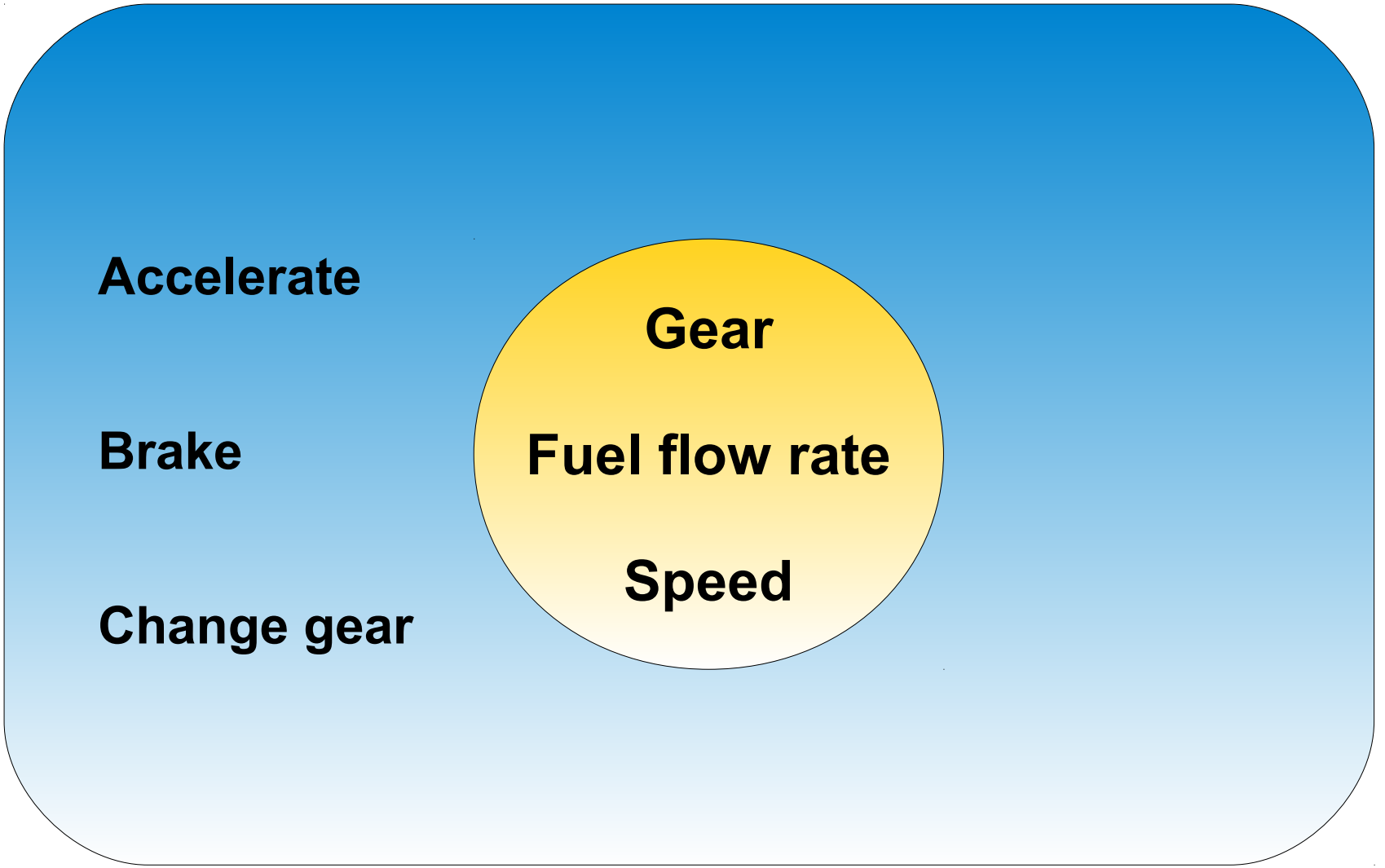
OOP

Programming paradigm that uses “objects”

Object

- **Data and methods**(functions/operations) encapsulated in an entity
- Data and methods are considered **members** of the object
- An object can access any of its members
- Outside code can only access a subset of the members

Objects



OOP

Classes

- "blueprint" for creating an object
- source code describes the class

Object

- Instance of a class
- Have common properties, attributes, operations, behavior with other objects of the same class

Class

```
class Point
{
        void setX(const int xval);
        void setY(const int yval);
        int getX();
        int getY();
        int _x, _y;
};

Point myLocation;
```

Protection Levels

```
class Point
{
public:
        void setX(const int xval);
        void setY(const int yval);
        int getX();
        int getY();
private:
        int _x, _y;
};

Point myLocation;
```

Class/Object

```
class Point
{
public:
        void setX(const int xval);
        void setY(const int yval);
        int getX();
        int getY();
private:
        int _x, _y;
};
```

Class Prototype/Declaration

```
Point myLocation;
```

Object

Members

```
class Point
{
public:
        void setX(const int xval);
        void setY(const int yval);
        int getX();
        int getY();
private:
        int _x, _y;
};
```

Member Functions

```
void setX(const int xval);
```

Member Functions

```
void setX(const int xval);


void Point::setX(const int xval)
{
        If (xval >=0)
                _x = xval;
}
```

Member Functions

Accessing members from outside the class

```
Point myLocation;

myLocation.setX(5);

myLocation._x = -1;
```

Member Data

What happens if we don't initialize member data?

Member Data

Constructor(s)

- Special member function that allows initialization of data
- Has same name as the class and has no return type
- Called automatically when an object is instantiated
- Cannot be called explicitly on an object
- Every class will have a constructor, if one is not supplied by the programmer, the compiler will create one

Constructor

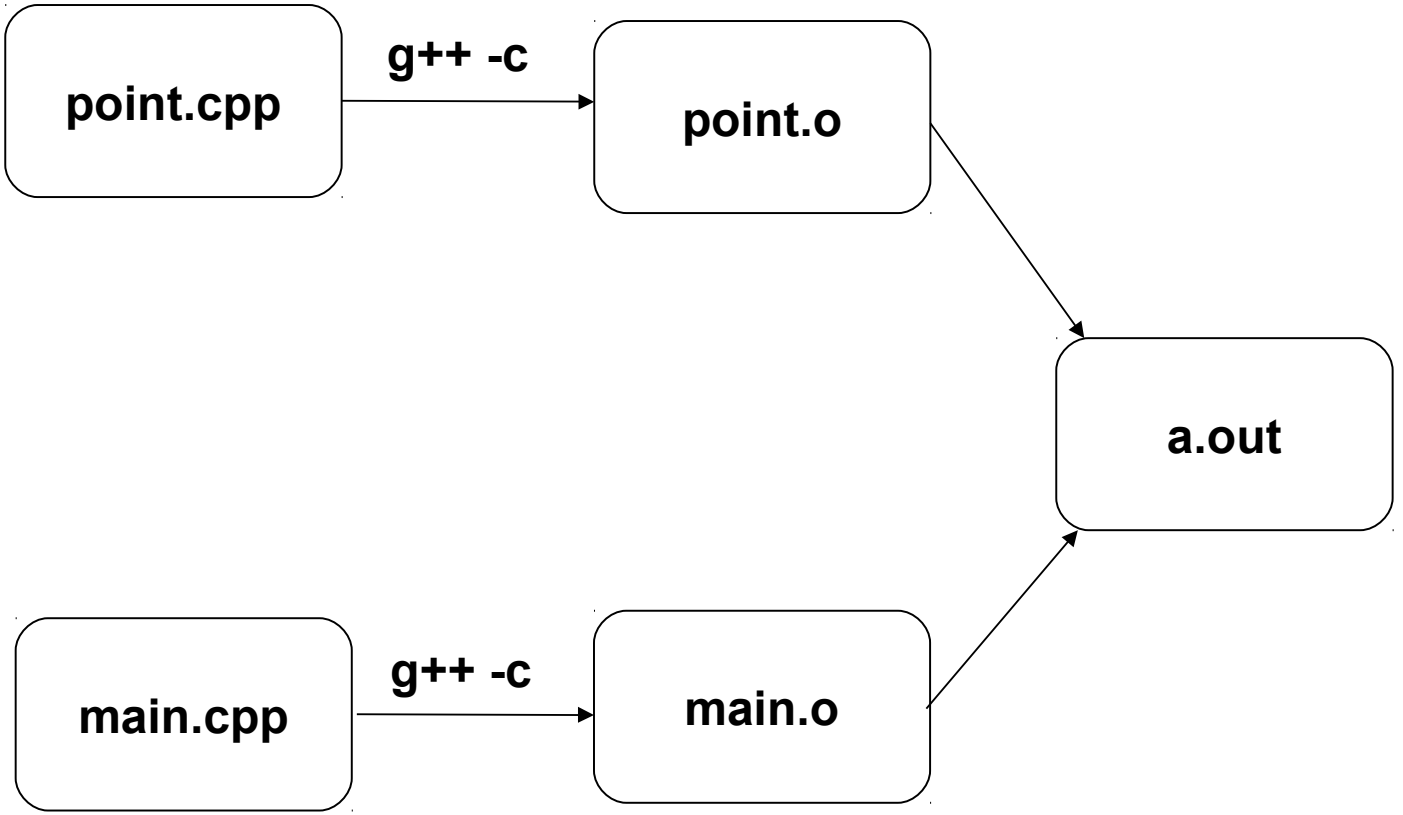
```
Point::Point()
{
        _x = 0;
        _y = 0;
}
```

Constructor

```
Point::Point(int x, int y)
{
        _x = x;
        _y = y;
}
```

Definitions/Declarations

Definitions/Declarations



Class Prototype

```
#include “point.h”

Point::Point(int x, int y)
{
        _x = x;
        _y = y;
}
void Point::setX(const int xval)
{
        If (xval >=0)
        _x = xval;
}
```