

today:

- simple classes
- FINAL EXAM: will be on MONDAY MAY 21, 1.00pm–3.00pm (room to be announced...)

simple classes

- classes are ways of organizing programs to provide structure
- a class is a special kind of *compound* data type
- classes are compound because they have *members*
- there are two types of members in classes:
 - *data* members
 - *function* members
- the *dot operator* (.) is used to indicate the member of a class
- you have already used three classes this semester:
 - `ifstream`
 - `ofstream`
 - `string`
- can you think of some of the member functions that belong to these classes?

- here are some of the member functions that belong to these classes:
 - `ifstream`:
 - * `open()`, `close()`, `eof()`
 - `ofstream`:
 - * `open()`, `close()`
 - `string`:
 - * `length()`, `clear()`, `erase()`, `replace()`, `insert()`, `find()`, `substr()`
- we have also mentioned a few data members, though all of these are actually constants and so are treated somewhat different from data variables (which we'll talk about later):
 - `ios::in`, `ios::out` — these belong to the `ios` class (`ifstream` and `ofstream` are created based on the `ios` class)
 - `string::npos`

- we use these classes by declaring variables whose data type is one of these classes, e.g.:
`string x;`
- we call `x` an *object* of type `string`
- then we can use the `string` member functions to operate on the object `x`, e.g.:

`string x;`
`x.clear();`
`x.insert( 0, "hello" );`

Notice the `x.` ("x dot") notation

simple class example

- suppose we wanted to create a program that contains the address book from your cell phone
- look at your cell phone address book:
 - what kind of information is listed for each entry?
 - for example:
 - * name (first name and last name)
 - * cell phone number
 - * email address
 - * home phone number
 - * work phone number
- these are called *fields*
- if we wanted to write a program that stored all this information for everyone in our cell phone address book, we could do something like this (we'll pretend we only have 3 friends...):

```
/**
 * p1.cpp
 * 07-may-2007/sklar
 *
 * this program motivates the use of simple classes in C++.
 *
 */

#include <iostream>
using namespace std;

void readData( string &last_name, string &first_name,
               string &cell_number, string &email, string &home_number,
               string &work_number, int &birth_day, int &birth_month,
               int &birth_year ) {
    cout << "enter last name: ";
    cin >> last_name;
    cout << "enter first name: ";
    cin >> first_name;
```

```
    cout << "enter cell number: ";
    cin >> cell_number;
    cout << "enter email: ";
    cin >> email;
    cout << "enter home number: ";
    cin >> home_number;
    cout << "enter work number: ";
    cin >> work_number;
    cout << "enter birthday (DD MM YY): ";
    cin >> birth_day;
    cin >> birth_month;
    cin >> birth_year;
    cout << "thanks!" << endl;
} // end of readData()
```

```
void writeData( string last_name, string first_name,
               string cell_number, string email, string home_number,
               string work_number, int birth_day, int birth_month,
               int birth_year ) {
```

```
    cout << "name: " << first_name << " " << last_name << endl;
    cout << "phone: " << cell_number << " (C)\n";
    cout << "      " << home_number << " (H)\n";
    cout << "      " << work_number << " (W)\n";
    cout << "email: " << email << endl;
    cout << "birthday: " << birth_day << "/" << birth_month << "/" <<
        birth_year << endl;
} // end of writeData()
```

```
int main() {
    string last_name[3];
    string first_name[3];
    string cell_number[3];
    string email[3];
    string home_number[3];
    string work_number[3];
    int birth_day[3];
    int birth_month[3];
    int birth_year[3];
```

```

cout << "enter data for 3 people...\n";
for ( int i=0; i<3; i++ ) {
    readData( last_name[i], first_name[i], cell_number[i], email[i],
              home_number[i], work_number[i], birth_day[i],
              birth_month[i], birth_year[i] );
}

cout << "here are all the people...\n";
for ( i=0; i<3; i++ ) {
    writeData( last_name[i], first_name[i], cell_number[i], email[i],
              home_number[i], work_number[i], birth_day[i],
              birth_month[i], birth_year[i] );
}

} // end of main()

```

- the idea is that it is annoying to have to keep track of so many parallel arrays
- so this is where we introduce a *class*
- a class will help us link together all the fields for each entry in the cell phone book
- here is a definition of a class that can hold such an entry:

```

class person {
public:
    string last_name;
    string first_name;
    string cell_number;
    string email;
    string home_number;
    string work_number;
    int birth_day;
    int birth_month;
    int birth_year;
};

```

- things to notice:
 - two new C++ keywords: `class` and `public`
 - there is a semi-colon at the END OF THE CLASS DEFINITION, after the last curly brace `{}`
- now here is our example (above) re-written using this simple class (but for only one person—next, we'll show how to do it with more than one person):

```

/**
 * p2.cpp
 * 07-may-2007/sklar
 *
 * this program demonstrates the use of simple classes in C++.
 *
 */

#include <iostream>
using namespace std;

class person {
public:

```

```

    string last_name;
    string first_name;
    string cell_number;
    string email;
    string home_number;
    string work_number;
    int birth_day;
    int birth_month;
    int birth_year;
};

void readData( person &p ) {
    cout << "enter last name: ";
    cin >> p.last_name;
    cout << "enter first name: ";
    cin >> p.first_name;
    cout << "enter cell number: ";
    cin >> p.cell_number;
    cout << "enter email: ";

```

```

    cin >> p.email;
    cout << "enter home number: ";
    cin >> p.home_number;
    cout << "enter work number: ";
    cin >> p.work_number;
    cout << "enter birthday (DD MM YY): ";
    cin >> p.birth_day;
    cin >> p.birth_month;
    cin >> p.birth_year;
    cout << "thanks!" << endl;
} // end of readData()

void writeData( person p ) {
    cout << "name: " << p.first_name << " " << p.last_name << endl;
    cout << "phone: " << p.cell_number << " (C)\n";
    cout << "      " << p.home_number << " (H)\n";
    cout << "      " << p.work_number << " (W)\n";
    cout << "email: " << p.email << endl;
    cout << "birthday: " << p.birth_day << "/" << p.birth_month << "/" <<

```

```

        p.birth_year << endl;
    } // end of writeData()

int main() {
    person p;
    readData( p );
    writeData( p );
} // end of main()

```

arrays of objects

- you can declare an array of a class
- so that each element in the array is an object of that class
- here is our example again, but with an array of person objects:

```

/**
 * p4.cpp
 * 07-may-2007/sklar
 *
 * this program demonstrates arrays of simple classes in C++.
 *
 */

#include <iostream>
using namespace std;

class name {
public:

```

```

    string last;
    string first;
};

class person {
public:
    name my_name;
    string cell_number;
    string email;
    string home_number;
    string work_number;
    int birth_day;
    int birth_month;
    int birth_year;
};

void readData( person &p ) {
    cout << "enter last name: ";
    cin >> p.my_name.last;

```

```

cout << "enter first name: ";
cin >> p.my_name.first;
cout << "enter cell number: ";
cin >> p.cell_number;
cout << "enter email: ";
cin >> p.email;
cout << "enter home number: ";
cin >> p.home_number;
cout << "enter work number: ";
cin >> p.work_number;
cout << "enter birthday (DD MM YY): ";
cin >> p.birth_day;
cin >> p.birth_month;
cin >> p.birth_year;
cout << "thanks!" << endl;
} // end of readData()

void writeData( person p ) {
    cout << "name: " << p.my_name.first << " " << p.my_name.last << endl;

```

```

        cout << "phone: " << p.cell_number << " (C)\n";
        cout << "        " << p.home_number << " (H)\n";
        cout << "        " << p.work_number << " (W)\n";
        cout << "email: " << p.email << endl;
        cout << "birthday: " << p.birth_day << "/" << p.birth_month << "/" <<
            p.birth_year << endl;
    } // end of writeData()

int main() {
    person p[3];
    int i;
    cout << "enter data for 3 people...\n";
    for ( i=0; i<3; i++ ) {
        readData( p[i] );
    }
    cout << "here are all the people...\n";
    for ( i=0; i<3; i++ ) {
        writeData( p[i] );
    }
}

```

```

    cout << "here are just the people's names...\n";
    for ( i=0; i<3; i++ ) {
        cout << p[i].my_name.first << " " << p[i].my_name.last << endl;
    }
} // end of main()

```

nested classes

- finally, you can *nest* classes, which means declare a data member in one class whose data type is that of another class
- so here is a modified version of the one-person address book, using two classes:

```

/**
 * p3.cpp
 * 07-may-2007/sklar
 *
 * this program demonstrates the use of nested simple classes in C++.
 *
 */

#include <iostream>
using namespace std;

class name {
public:

```

```

    string last;
    string first;
};

```

```

class person {
public:
    name my_name;
    string cell_number;
    string email;
    string home_number;
    string work_number;
    int birth_day;
    int birth_month;
    int birth_year;
};

```

```

void readData( person &p ) {
    cout << "enter last name: ";
    cin >> p.my_name.last;
}

```

```

    cout << "enter first name: ";
    cin >> p.my_name.first;
    cout << "enter cell number: ";
    cin >> p.cell_number;
    cout << "enter email: ";
    cin >> p.email;
    cout << "enter home number: ";
    cin >> p.home_number;
    cout << "enter work number: ";
    cin >> p.work_number;
    cout << "enter birthday (DD MM YY): ";
    cin >> p.birth_day;
    cin >> p.birth_month;
    cin >> p.birth_year;
    cout << "thanks!" << endl;
} // end of readData()

```

```

void writeData( person p ) {
    cout << "name:  " << p.my_name.first << " " << p.my_name.last << endl;
}

```

```

    cout << "phone: " << p.cell_number << " (C)\n";
    cout << "      " << p.home_number << " (H)\n";
    cout << "      " << p.work_number << " (W)\n";
    cout << "email: " << p.email << endl;
    cout << "birthday: " << p.birth_day << "/" << p.birth_month << "/" <<
        p.birth_year << endl;
} // end of writeData()

```

```

int main() {
    person p;
    readData( p );
    writeData( p );
} // end of main()

```