Object Oriented Programming in C++

- Represents the concept of “objects”
- Problem solving
  • 20 projects
  • 6 quizzes
Log on to the machine

Username:
- first 2 letters of your last name followed by
- first 2 letters of your first name followed by
- last 4 digits of your CUNYfirst ID
  e.g. yaka5678

Password:
- 8 digits of your CUNYfirst ID
  e.g. 12345678
SSH

Venus server
Available on my lab website.

Mac user:
Use the Terminal Program and command
ssh your_login@venus.cs.qc.cuny.edu

you_login: same as the log in name for the machine in the lab.
Microsoft Visual C++

- support many debug techniques (checkpoints, error detection)
- faster to debug
- more detailed error message
- format code (Edit → Format Selection)
Homework submission

Subject headline: CS211 follow by firstname lastname and homework number

All homework should be submitted in an organized and well-formatted way

All homework should be able to COMPILE and Run.

If you have any difficulty or question regarding the homework, please see me during my office hour:
Tuesday and Thursday: 5:00PM – 5:30PM in room A201
Review

- if statement
- compound boolean
- for loop
- while loop
- array
- 2D array
- function
- post-increment and pre-increment
- recursive function
If Statement

```java
if (boolean_value_condition_check) {
    body statement if the condition is true ...
}
else if (check_here_if_the_first_condition_is_false)
    body statement if this condition is true
else
    body statement - if non of the previous condition is true
```
Boolean

- true or false value
- a comparison

Example:
```java
int i = 0;
bool check = i > 0;
bool even = i % 2 == 0;
```

Compound Boolean

- combine two boolean value
- || combined with or
- && combined with and

Example:
```java
int num, age;
bool minor = age < 18 && age >= 16;
bool notTwoDigits = num<10 || num>=100;
```
Post-Increment

```c
int i = 0;
i++;
```

Pre-Increment

```c
int i = 0;
++i;
```

eexample

```c
int i = 0;
cout << ++i;
i = 0;
cout << i++;;
```
While Loop

while (repeat_when_condition_is_true) {
    body statement ....
}

example

int i = 0;
while (i++ < 10) cout << i;
int i = 0;
while (i < 10) cout << i;
Array

- a collection of data
- same data type
- size of the collection
- index of the entry

example

```c
int array[100];

3 6 8

2

cout << array[0];

cout << array[99];
```

2-D Array

-a collection of data
-data type is an array
-usually visualize it as a table
-with row and column indices

element

```c
int table[10][10];

5 ... 9
...
...
...
0 ... 3
```

```c
cout << table[0][0];
cout << table[0][9];
```
Function

- a group of code that carries a single task

- a big task may break into several smaller task

- may be called from another function
Function

- a group of code that carries a single task
- a big task may break into several smaller tasks
- may be called from another function
Function

```cpp
int main(){
    int a = 3, b = 4;
    cout << (a+b)/2;
    cin >> a >> b;
    cout << (a+b)/2;
    return 0;
}
```

```cpp
int avg(int, int);
int main(){
    int a = 3, b = 4;
    cout << avg(a,b);
    cin >> a >> b;
    cout << avg(a,b);
    return 0;
}
```

```cpp
int avg(int f, int s){
    return (f+s)/2;
}
```
Recursive Function

-a function that calls itself to finish a repeated task

-ask itself to do a smaller portion of the task (recursive call)

-finish the task base on the result of the smaller portion (iterative step)

-includes a situation when the task is a single process that cannot be break down as repeated task (base case)
Recursive Function

Example:

```c
void printArray(int a[], int size){
    if (size == 0) return;
    printArray(a, size - 1);
    cout << a[size - 1];
}
```