Structure of C++ Programs

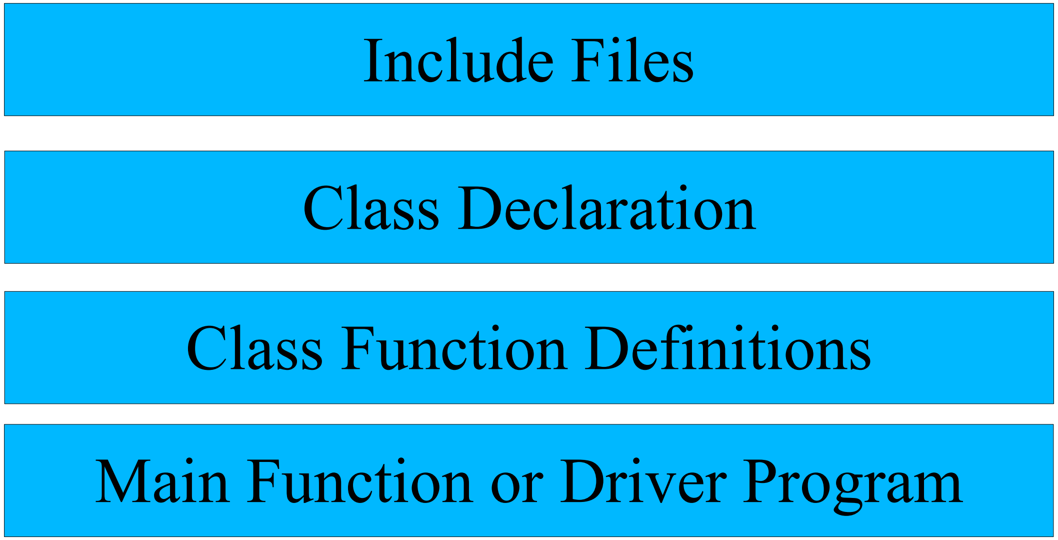
Part 3

Dr. Vipin Bhatnagar
Dept. of Physics

M.Sc. Medical Physics, Nuclear Medicine Course - 2007

Program Layout

- Generic Structure



Include Files

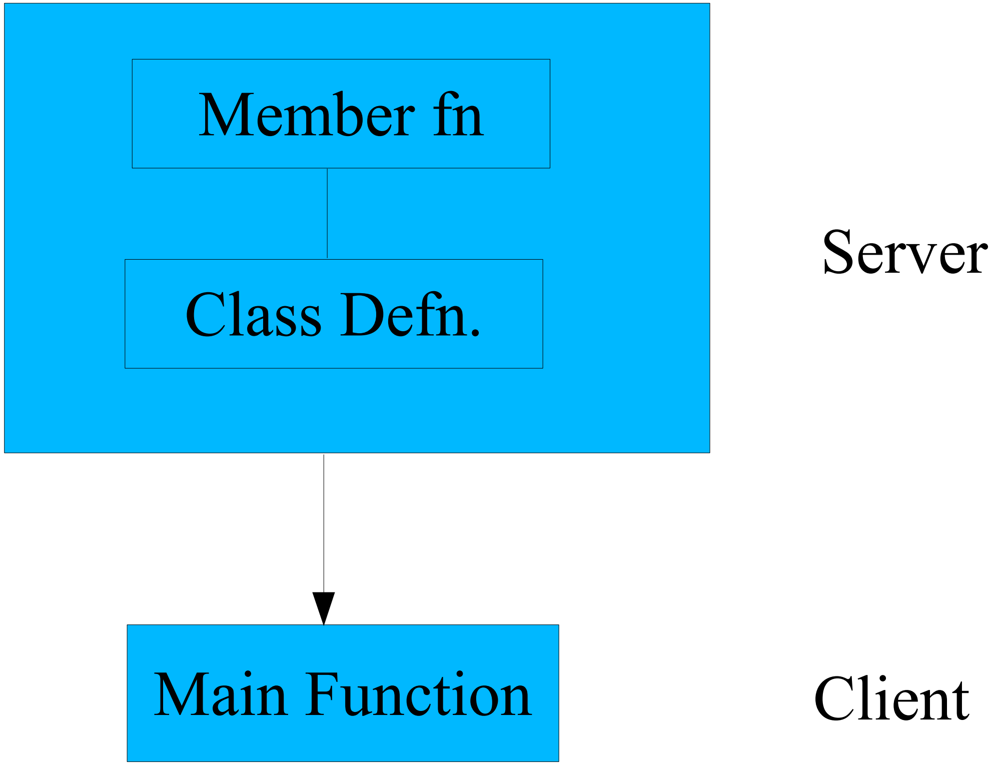
Class Declaration

Class Function Definitions

Main Function or Driver Program

Client- Server Model

- Similar to Internet Web Server + Browser model

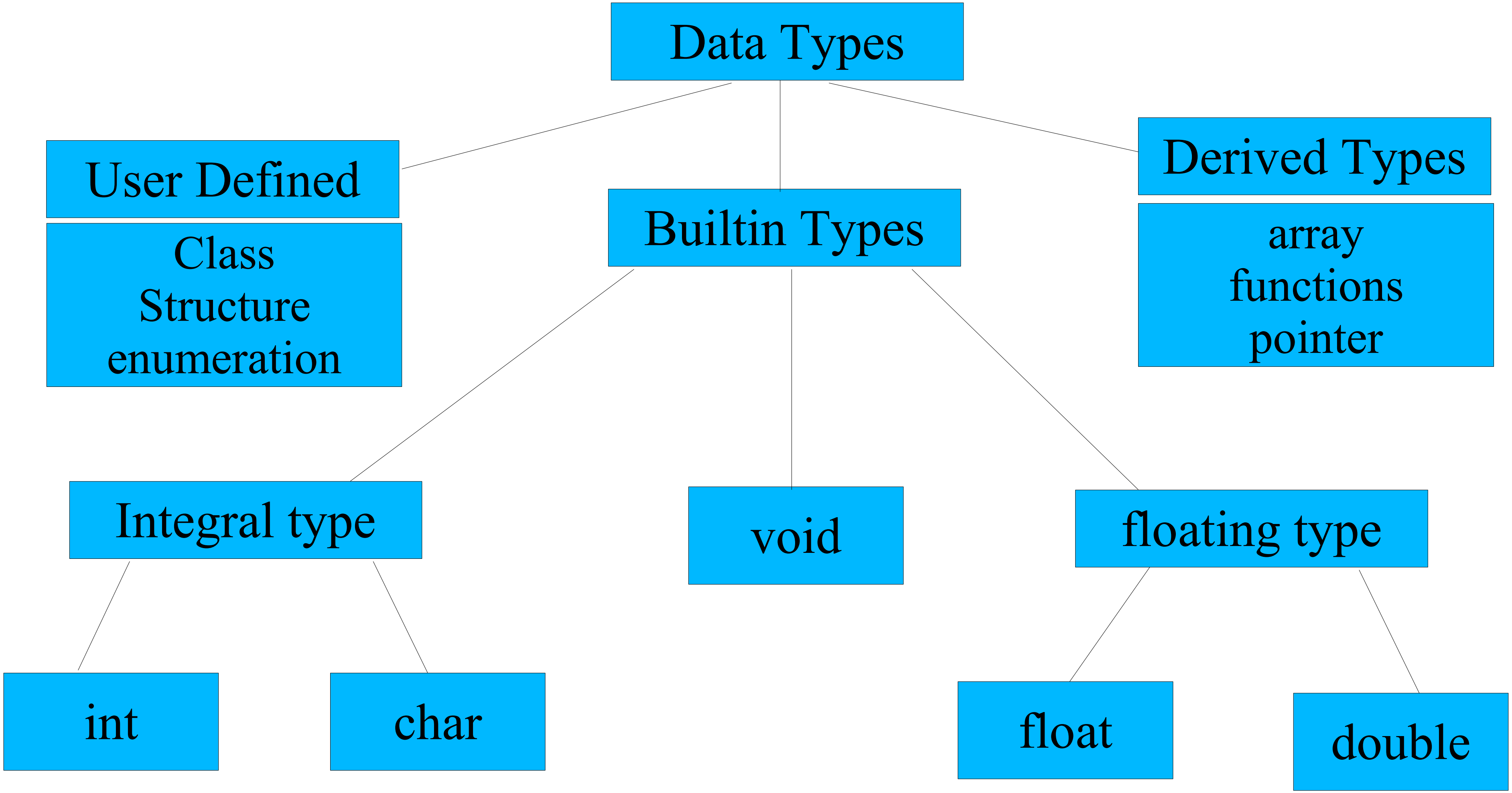


Tokens: Smallest individual entity in C++

- Keywords: *reserved names*
  - *class, int, char, main, double, const, etc.*
- Identifiers: *names of variables, functions, arrays, classes, etc.*
  - *alphabetic characters, digits & underscore (_), e.g. my_name, student2, ResultMSc2*
  - *cannot start with digit: 5name* ✗
  - *UPPERCASE & lowercase are Different*
  - *A keyword cannot be used as a variable name,*

eg. int char (not allowed)

Basic Data Types



Size & Range of C++ basic data types

char	1 byte	-128 to 127
int	2 bytes	-32768 to 32767
float	4 bytes	3.4E-38 - 3.4E+38
double	8 bytes	1.7E-308 – 1.7E+308

- unsigned char, short int, long int, etc...

Example 3:

```
int main()
{
   float x;
   float sum = 0;
   for (int i = 1; i < 5; i++)
   {
      cin >> x;
      sum = sum + x;
   }
   float average;
   average = sum/i;
   cout << average;
   return(0);
}
```

Derived Data Types

- Arrays: *char name[30]; char string[4] = “xyz”;*

The number in [] has one additional space for the null character \0.

- Functions: see second example (more details later)

- Pointers: *Used in C++ for memory mgmt. And achieving polymorphism. (come from C)*

As the name suggests, a ptr. can point to any variable of correct type.

Pointers usage eg.

```
int *ip;    // int pointer
ip = &x;    // address of x assigned to ip
*ip = 10;   // 10 assigned to x thro'
  indirection
```

*Use the above to print the value of x.

hint: use main function and cout object with << operator

Control Structures

All functioning algorithm can be coded using 3 types of basic control structures:

- Sequence structure
- Selection structure
- Loop structure

All the above implement ***one-entry***, ***one-exit*** concept

(this approach comes frrom modular programming)

And using one or more of above in programming is known as ***structured programming***

Selection structure

- **I:**

```
if (expression is true)
  {
     action1;
  }
action2;
```

- **II:**

```
if (exp is true)
  { action1;}
else
  { action2;}
action3;
```