# An Introduction to Programming through C++

Abhiram G. Ranade

Lecture sequence 2.1

Ch. 3: Variables and Data Types

## Outline

How to perform some basic operations needed in all programs

- Store numbers in the memory of a computer.
- Read numbers into memory from the keyboard.
- Print numbers on the screen.
- Perform arithmetic.

Some programs based on all this.

## Reserving memory for storing numbers

- Before you store numbers, you must explicitly reserve space for storing them.
  - “space” : region of memory
- This is done by a “variable definition” statement.
- variable: name given to the space you reserved.
  - “Value of a variable”: value stored in the variable
- You must also state what kind of values will be stored in the variable: “data type” of the variable.

## Variable creation/definition

Statement form:

***data-type-name variable-name;***

- Example from chapter 1:

***int nsides;***

- ***int*** : data type name. Short for “integer”.
  - Reserve space for storing integer values, positive or negative, of a “standard” size.
  - Standard size = 32 bits on most computers.
  - Two’s complement representation will typically be used.
- ***nsides*** : name given to reserved space, or the created variable.

# Variable names: “Identifiers”

- Sequence of 1 or more letters, digits and the underscore “_” character
  - Should not begin with a digit
  - Some words such as ***int*** cannot be used as variable names. Reserved by C++ for its own use.
  - case matters. ***ABC*** and ***abc*** are distinct identifiers
  - Space not allowed inside variable name
- Examples: ***nsides, telephone_number, x, x123, third_cousin***
- Non-examples: ***#sides, 3rd_cousin, 3 rd cousin***
- Recommendation: use meaningful names, describing the purpose for which the variable will be used.

## Some other data types of C++

- ***unsigned int*** : Used for storing integers which will always be positive.
  - 1 word will be allocated.
  - Ordinary binary representation will be used.
- ***char*** : Used for storing characters or small integers.
  - 1 byte will be allocated.
  - ASCII code of characters is stored.
- ***float*** : Used for storing real numbers
  - 1 word will be allocated.
  - IEEE FP representation, 8 bits exponent, 24 bits significand.
- ***double*** : Used for storing real numbers
  - 2 words will be allocated.
  - IEEE FP representation, 11 bits exponent, 53 bits significand.

## Examples

***unsigned int telephone_number;***
***float mass, acceleration;***

- OK to define several variables in same statement.
- Keyword ***long*** : says, “I need to store bigger or more precise numbers, so give me more than usual space.”

***long unsigned int cryptographic_password;***

- Likely 64 bits will be allocated.

***long double more_precise_acceleration;***

- Likely 96 bits will be allocated

# Variable initialization

- A value can be stored in a variable at the time of creation

```
int i=0, result;
float vx=1.0, vy=2.0e5, weight;
```

- ***i, vx,vy*** given values as well as defined.
- 2.0e5 is how you write $2.0*10^5$
- Although the computer uses binary, you write in decimal.

```
char command = ‘f’;
```

- ***‘f’*** is a “character constant”. It represents the ASCII value of the quoted character.

## const

```
const double avogadro = 6.022e23;
```

- The keyword **const** : value assigned cannot be changed.

# Reading values into variables

- We did this in chapter 1:

```
cin >> nsides;
```

- Can read into several variables one after another

```
cin >> vx >> vy;
```

- User expected to type in values consistent with the type of the variable into which it is to be read.
  - "Whitespace" = space characters, tabs, newlines, typed by the user are ignored.
  - newline/enter key must be pressed after values are typed
- If you read into a **char** type variable, the ASCII code of the typed character gets stored.

```
char command;
cin >> command;
```

- If you type the character 'f', its ASCII value, 102, will get stored.

# Printing variables on the screen

- General form: **cout << variable-name;**
- To print newline, use **endl**.
- Additional text can be printed by enclosing it in quotes.
- Many things can be printed by placing **<<** between them.

```
cout <<"Position: "<< x << ",  " << y  << endl;
```

- Prints the text "Position: ", then values of variables **x, y** with a comma between them and a newline after them.
- If you print a **char** variable, then the content is interpreted as an ASCII code, and the corresponding character is printed.

```
char command = 'G', command2 = 97;
cout << command << command2;  // Ga will be printed.
```

- To force output to appear, print **endl**, or read something immediately.

## Exercises

- Create a ***double*** variables ***temperature, pressure***, with pressure initialized to 1.0
- Create a constant double variable ***PI*** initialized to 3.141592
- Create a char variable ***yOrNo*** initialized to the character ‘y’.
- What name would you give to a variable which is meant to store the number of students in a class?

## What we discussed

- How to define variables
- How to initialize variables
- How to read a value into a variable and print the value of a variable.