C Program Structure

Programming in C

Shirley B. Chu
shirley.chu@delasalle.ph

College of Computer Studies
De La Salle University

November 4, 2021

Writing C programs

What you need...

- editor such as Notepad++, or

Writing C programs

What you need...

- editor such as Notepad++, or
- compiler

Writing C programs

What you need...

- editor such as Notepad++, or
- compiler
- IDE (Integrated development environment); and

Writing C programs

What you need...

- editor such as Notepad++, or
- compiler
- IDE (Integrated development environment); and

Saving your source code...

Writing C programs

What you need...

- editor such as Notepad++, or
- compiler
- IDE (Integrated development environment); and

Saving your source code...

- file name without space/s

Writing C programs

What you need...

- editor such as Notepad++, or
- compiler
- IDE (Integrated development environment); and

Saving your source code...

- file name without space/s
- file extension: **.c**

Semicolon ;

- ; is used to mark the end of a statement, and the beginning of another statement.

Semicolon ;

- ; is used to mark the end of a statement, and the beginning of another statement.
- All C statements must end with a semi-colon.

Sample C Program

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

Comments

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

Comments

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);     /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- enclosed in /* and */

Comments

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);     /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- enclosed in /* and */
- starts with //

Comments

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);     /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- enclosed in /* and */
- starts with //
- can be placed anywhere in the program

Comments

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);     /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- enclosed in /* and */
- starts with //
- can be placed anywhere in the program
- are treated as white spaces by the compiler

Comments

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);     /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- enclosed in /* and */
- starts with //
- can be placed anywhere in the program
- are treated as white spaces by the compiler
- used to put notes

Comments

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);     /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- enclosed in /* and */
- starts with //
- can be placed anywhere in the program
- are treated as white spaces by the compiler
- used to put notes
- used to temporarily remove a statement or groups of statements

Comments

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);     /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- enclosed in /* and */
- starts with //
- can be placed anywhere in the program
- are treated as white spaces by the compiler
- used to put notes
- used to temporarily remove a statement or groups of statements
- **Introductory comment** : usually contains the name of the author/s, date, assumptions, and description of the program

Sample C Program

#include

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- Syntax:
  `#include<`*header_file*`>`

Sample C Program

#include

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- Syntax:
  `#include<`*header_file*`>`
- `#include`

Sample C Program

#include

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- Syntax:
  `#include<`*header_file*`>`
- `#include`
  - a preprocessor directive;

Sample C Program

#include

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- Syntax:
  `#include<`*header_file*`>`
- `#include`
  - a preprocessor directive;
  - compiler inserts the contents of the indicated file

Sample C Program

#include

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- Syntax:
  `#include<`*header_file*`>`
- `#include`
  - a preprocessor directive;
  - compiler inserts the contents of the indicated file
- *header_file*

Sample C Program

#include

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- Syntax:
  `#include<`*header_file*`>`
- `#include`
  - a preprocessor directive;
  - compiler inserts the contents of the indicated file
- *header_file*
  - file extension: **.h**

Sample C Program

#include

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- Syntax:
  `#include<`*header_file*`>`
- `#include`
  - a preprocessor directive;
  - compiler inserts the contents of the indicated file
- *header_file*
  - file extension: **.h**
  - contains definitions that are required to run the program

Sample C Program

#include

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- Syntax:
  `#include<`*header_file*`>`
- `#include`
  - a preprocessor directive;
  - compiler inserts the contents of the indicated file
- *header_file*
  - file extension: **.h**
  - contains definitions that are required to run the program
- All necessary header files must be explicitly included.

Sample C Program

#define

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- allows constant values to be declared and used throughout the code

Sample C Program

#define

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- allows constant values to be declared and used throughout the code
- value cannot be changed by the code in the program

Sample C Program

#define

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- allows constant values to be declared and used throughout the code
- value cannot be changed by the code in the program
- convention: identifiers are in all uppercase letters and may have underscores

Function definitions

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- subroutines created by the programmer

Function definitions

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- subroutines created by the programmer
- performs a specific task

main() Function

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- entry point of a C program

main() Function

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- entry point of a C program
- may contain:

main() Function

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- entry point of a C program
- may contain:
  - variable declarations;

main() Function

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- entry point of a C program
- may contain:
  - variable declarations;
  - statements;

main() Function

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- entry point of a C program
- may contain:
  - variable declarations;
  - statements;
  - function calls

main() Function

```
/* This is a sample program.  */

#include<stdio.h>

#define MSG "Hi There"
#define SIZE 50

void displayMessage ()
{
  printf (MSG);    /*displays Hi*/
}

int main ()
{
  int nVal = 5;
  displayMessage ();
  printf ("Number is %d\n", nVal);
  return 0;
}
```

- entry point of a C program
- may contain:
  - variable declarations;
  - statements;
  - function calls

Compiling and Running a C Program

1. Run Notepad++

Compiling and Running a C Program

1. Run Notepad++
2. Start typing.

Compiling and Running a C Program

1. Run Notepad++
2. Start typing.
3. Save your file as **MyFirstProgram.c**

Compiling and Running a C Program

1. Run Notepad++
2. Start typing.
3. Save your file as **MyFirstProgram.c**
4. Run the Command Prompt.

Compiling and Running a C Program

1. Run Notepad++
2. Start typing.
3. Save your file as **MyFirstProgram.c**
4. Run the Command Prompt.
5. Change directory to where your source code is saved.

Compiling and Running a C Program

1. Run Notepad++
2. Start typing.
3. Save your file as **MyFirstProgram.c**
4. Run the Command Prompt.
5. Change directory to where your source code is saved.
6. To compile your program, type

```
gcc -Wall MyFirstProgram.c -o MyFirstProgram
```

Compiling and Running a C Program

1. Run Notepad++
2. Start typing.
3. Save your file as **MyFirstProgram.c**
4. Run the Command Prompt.
5. Change directory to where your source code is saved.
6. To compile your program, type

   ```
   gcc -Wall MyFirstProgram.c -o MyFirstProgram
   ```

7. If there are errors, go back to your code and fix them. Compile again.

Compiling and Running a C Program

1. Run Notepad++
2. Start typing.
3. Save your file as **MyFirstProgram.c**
4. Run the Command Prompt.
5. Change directory to where your source code is saved.
6. To compile your program, type

   ```
   gcc -Wall MyFirstProgram.c -o MyFirstProgram
   ```

7. If there are errors, go back to your code and fix them. Compile again.
8. Once compilation is successful, i.e. no errors, run the program:

   ```
   MyFirstProgram
   ```

☺ Thank you! ☺