Féidearthachtaí as Cuimse
Infinite Possibilities

Week 3 - Tutorial

Programming - Week 3 – 2nd October 2024

OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH
TECHNOLOGICAL UNIVERSITY DUBLIN

Overview

- Casting
  - Implicit
  - Explicit
- Keyboard Input
- getchar and putchar
- Program Conventions
- Programming Pitfalls

Casting

- Casting in C is allows us to convert a variable from one data type to another.
- When performing operations with different data types that use different amounts of memory (different sizes)

Implicit Casting

- Also called type promotion
- The type promotion is automatically performed by the C compiler

Example1.c > ...

```c
#include <stdio.h>

int main() {
    int i = 10;
    double d = i; // Implicit casting from int to double

    printf("Integer: %d, Double: %f\n", i, d);

    return 0;
}
```

i is an integer

d is a double

When **i** is assigned to **d**, the value is automatically cast to a double (10.0).

Casting: Mixed data type expressions

- This is a type of Implicit Casting
- This occurs when compatible types are mixed in operations (mixed data type expressions)

Example2.c > ...

```c
#include <stdio.h>

int main() {
    int var1 = 13;
    float var2 = 2.3;
    float var3;

    var3 = var1 / var2;

    printf("Integer(var1): %d, Float(var2): %f, Float(var3): %f\n", var1, var2, var3);

    return 0;
}
```

var1 is an Integer
Var2 is a float
Var3 is a float

Dividing var1 by var2 is a mixed data type expression. The result will be converted to a float by the compiler.

Explicit Casting

- Explicit casting is when you manually convert a variable from one type to another. This is a specific declaration in our C code.

- This is done using the following syntax:
  - (new_type) value
- Eg:

```
double d = 9.7;

int i = (int) d; // Explicitly cast double to int
```

Explicit Casting (example 1)

OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH
TECHNOLOGICAL UNIVERSITY DUBLIN

- Casting an Integer to a Double

Example3.c > ...

```c
#include <stdio.h>

int main() {
    int var1 = 12;
    double var2 = (double) var1; // Explicitly cast int to double

    printf("Integer: %d, Double after cast: %f\n", var1, var2);

    return 0;
}
```

Explicit Casting (example 2)

- Casting a float to an integer

Example4.c > ...

```c
#include <stdio.h>

int main() {
    float var1 = 12.7;
    int var2 = (int) var1; // Explicitly cast float to int

    printf("Float: %f, Integer after cast: %d\n", var1, var2);

    return 0;
}
```

Caution
The value stored in the Float will change when it is cast to an Integer

Keyboard Input

- In our C programs we may want the user to enter some data.
- Eg. Ask a user to enter their age.

- To get the users keyboard input, we use the scanf() function

- scanf() reads input from the user
- You must place the & character before the variable name is a scanf() statement

Why do we need the & for scanf

OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH
TECHNOLOGICAL UNIVERSITY DUBLIN

- When using the scanf() function, the ampersand (&) is used to pass the address of a variable to the scanf function.
- This is necessary because scanf() needs to modify the value of the variable where the input will be stored
- To to this it needs the memory address of the variable.
- Adding the & to the variable name lets us reference the memory where the variable stores its data.

scanf – Example 1

- Ask a user to input their age. Then display their age.

Example5.c > ...

```c
#include <stdio.h>

int main(void)
{
    int age = 0;

    printf("Please enter your age: \n");

    // Ensure you include the & character before the variable name
    scanf("%d", &age);

    printf("You are %d years old\n", age);

    return 0;
}
```

scanf – Example 2

- Addition Calculator

Example5.c > ...

```c
#include <stdio.h>

int main(void)
{
    float num1;
    float num2;
    float answer;

    printf("My Calculator..... \n");

    printf("Enter the first number:\n ");
    scanf("%f", &num1);

    printf("Enter the second number:\n ");
    scanf("%f", &num2);

    answer = num1 + num2;

    printf("%f + %f = %f \n", num1, num2, answer);

    return 0;
}
```

scanf – Example 3

```c
#include <stdio.h>
int main(void)
{
    int var = 0;
    char my_char = ' ';

    printf("Enter a single number and a single character\n");
    scanf("%d", &var);

    //while(getchar() != '\n');

    printf("Ente a char: \n");
    scanf(" %c", &my_char);

    printf("You entered %d and %c\n", var, my_char);

    return 0;
}
```

The space before %c tells scanf() to ignore any preceding whitespace characters, which can include newlines or spaces that may have been left in the input buffer. This ensures that scanf() reads the actual character input.

Using getchar() Instead of scanf()
ch = getchar(); // Reads a single character
printf("You entered: '%c'\n", ch);

Precision Specification

- To specify a floating-point number to x decimal places, you can do the following:

Example7.c > ...

```c
#include <stdio.h>

int main() {
    float var = 0;

    printf("Enter any float number\n");
    scanf("%f", &var);

    // Display the number entered correct to 2 decimal places
    printf("You entered %.2f", var);

    return 0;
}
```

getchar() and putchar()

OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH
TECHNOLOGICAL UNIVERSITY DUBLIN

C Example8.c > ...

```c
#include <stdio.h>

int main(void)
{

    char my_char = ' '; printf("Enter any character\n");

    // use the getchar() function
    //scanf("%c", &my_char);

    // same as above
    my_char = getchar();

    //printf("my_char contains %c", my_char);
    printf("\nYou entered ");
    putchar(my_char);

    return 0;
}
```

Program Conventions

- Comment at the start of the program + code
- Proper indentation
- Use of white space
- Naming Variables correctly

Programming Pitfall

- Don't forget to include the & in the scanf() statement

- Regarding precision formatting, only the printf() statement allows this to be done.
  - e.g., printf("you entered %.2f", num). You cannot place the precision formatting inside a scanf() statement. e.g., the following is wrong: scanf("%.2f", &num);

Questions