Introduction to C

Hongyi Zhang

CSC2100 Data Structures Tutorial 1

Information

- Course Information:
- Web Page:
  - http://www.cse.cuhk.edu.hk/irwin.king/teaching/csci2100/2015
- Tutorial Page:
  - http://www.cse.cuhk.edu.hk/irwin.king/teaching/csci2100/2015/tutorial
- Anti-plagiarism Policy:
  - http://www.cuhk.edu.hk/policy/academichonesty/

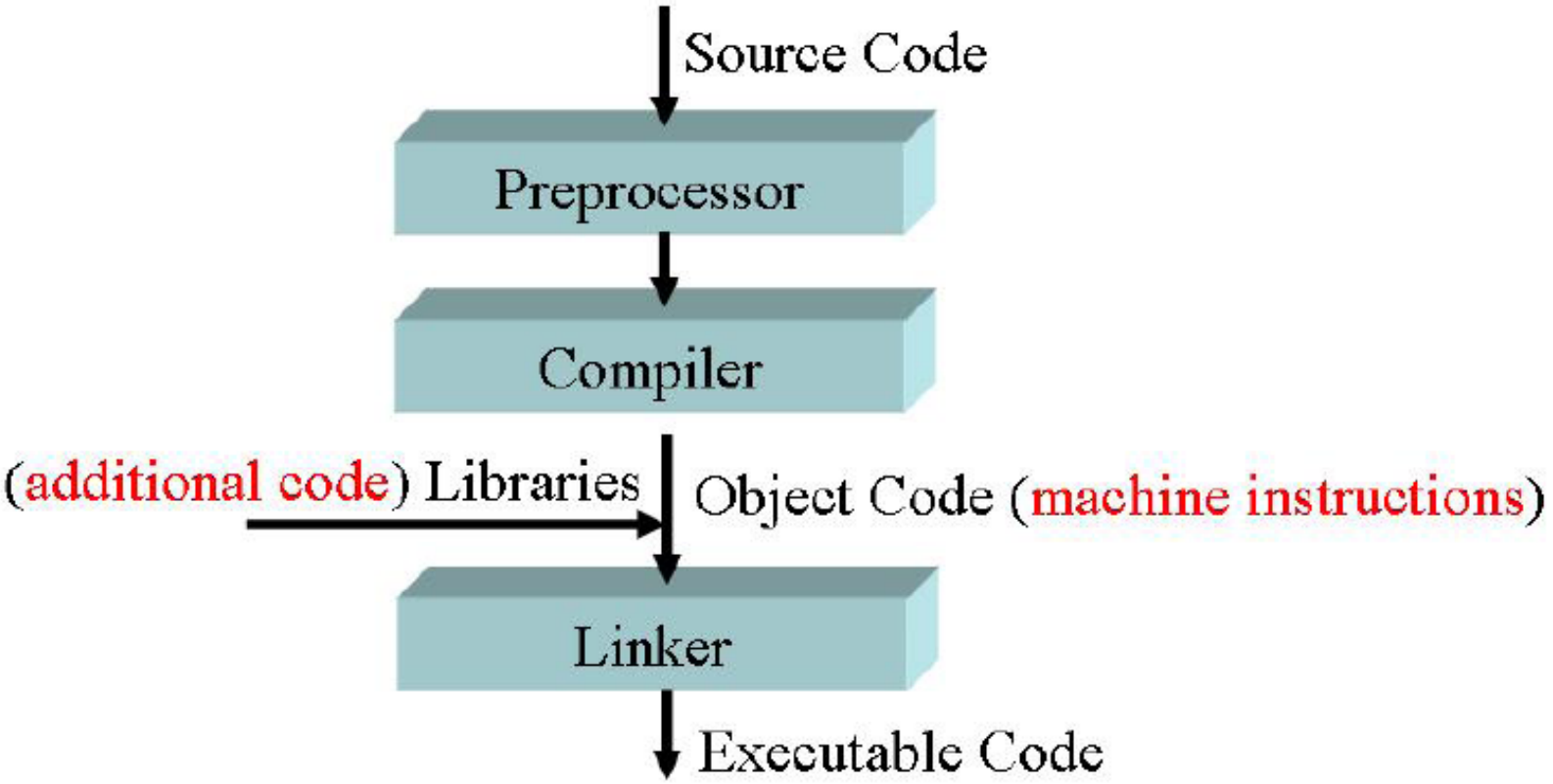
Information

- Assignment
  - There will be both written and programming parts in assignments.
    - Written part: submit to the assignment box in 10/F SHB.
    - Programming part: via Online Judge systems. (Will be introduced next week)
  - For non-CSE student, you will receive your login Id for CSC2100 online judge via your CUHK Link email account (after add/drop week).
    - Keep it safe and do not disclose it.

Introduction to C

- Basics
- If Statement
- Loops
- Functions
- Switch case
- Pointers
- Structures
- File I/O

The C Compilation Model



Introduction to C: Basics

```
/*a simple program
that has variables*/
#include <stdio.h>
int main()
{
   int x; //(32 bits)
   char y; //(8 bits)
   float z; //(32 bits)
   double t; //(64 bits)
   printf("hello world…\n");
   test = 1; //wrong, The variable declaration must appear first
   return 0;
}
```

Introduction to C: Basics

```
//reading input from console
#include <stdio.h>
int main()
{
    int num1;
    int num2;
    printf( "Please enter two numbers: " );
    scanf( "%d %d", &num1,&num2 );
    printf( "You entered %d %d", num1, num2 );
    return 0;
}
```

Introduction to C: if statement

```
#include <stdio.h>
int main()
{
    int age;                    /* Need a variable... */
    printf( "Please enter your age" );  /* Asks for age */
    scanf( "%d", &age );            /* The input is put in age */
    if ( age < 100 )
    {           /* If the age is less than 100 */
        printf ("You are pretty young!\n" ); /* Just to show you it works... */
    }
    else if ( age == 100 )
    {       /* I use else just to show an example */
        printf( "You are old\n" );
    }
    else
    {
        printf( "You are really old\n" );    /* Executed if no other statement is*/
    }
    return 0;
}
```

Introduction to C: Loops(for)

```
#include <stdio.h>
int main()
{
   int x;
   /* The loop goes while x < 10, and x increases by one every loop*/
   for ( x = 0; x < 10; x++ )
   {
      /* Keep in mind that the loop condition checks
         the conditional statement before it loops again.
         consequently, when x equals 10 the loop breaks.
         x is updated before the condition is checked. */
         printf( "%d\n", x );
   }
   return 0;
}
```

Introduction to C: Loops(while)

```
#include <stdio.h>
int main()
{
  int x = 0;  /* Don't forget to declare variables */
  while ( x < 10 )
  {   /* While x is less than 10 */
      printf( "%d\n", x );
      x++;    /* Update x so the condition can be met eventually */
  }
  return 0;
}
```

Introduction to C: Loops(do while)

```
#include <stdio.h>
int main()
{
  int x;
  x = 0;
  do
  {
    /* "Hello, world!" is printed at least one time
      even though the condition is false*/
     printf( "%d\n", x );
      x++;
  } while ( x != 10 );
  return 0;
}
```

Introduction to C: Loops(break and continue)

```
#include <stdio.h>
int main()
{
  int x;
  for(x=0;x<10;x++)
  {
        if(x==5)
        {
              break;
        }
        printf("%d\n",x);
  }
  return 0;
}
```

```
0
1
2
3
4
```

```
#include <stdio.h>
int main()
{
  int x;
  for(x=0;x<10;x++)
  {
        if(x==5)
        {
              continue;
        }
        printf("%d\n",x);
  }
  return 0;
}
```

```
0
1
2
3
4
6
7
8
9
```

```c
#include <stdio.h>
//function declaration, need to define the function body in other places
void playgame();
void loadgame();
void playmultiplayer();
int main()
{
   int input;
   printf( "1. Play game\n" );
   printf( "2. Load game\n" );
   printf( "3. Play multiplayer\n" );
   printf( "4. Exit\n" );
   printf( "Selection: " );
   scanf( "%d", &input );
   switch ( input ) {
      case 1:          /* Note the colon, not a semicolon */
         playgame();
         break;        //don't forget the break in each case
      case 2:
         loadgame();
         break;
      case 3:
         playmultiplayer();
         break;
      case 4:
         printf( "Thanks for playing!\n" );
         break;
      default:
         printf( "Bad input, quitting!\n" );
         break;
   }
   return 0;
}
```

switch

case

Introduction to C: function

```
#include <stdio.h>
//function declaration
int mult ( int x, int y );
int main()
{
  int x, y;
  printf( "Please input two numbers to be multiplied: " );
  scanf( "%d", &x );
  scanf( "%d", &y );
  printf( "The product of your two numbers is %d\n", mult( x, y ) );
  return 0;
}
//define the function body
//return value: int
//utility: return the multiplication of two integer values
//parameters: take two int parameters
int mult (int x, int y)
{
  return x * y;
}
```

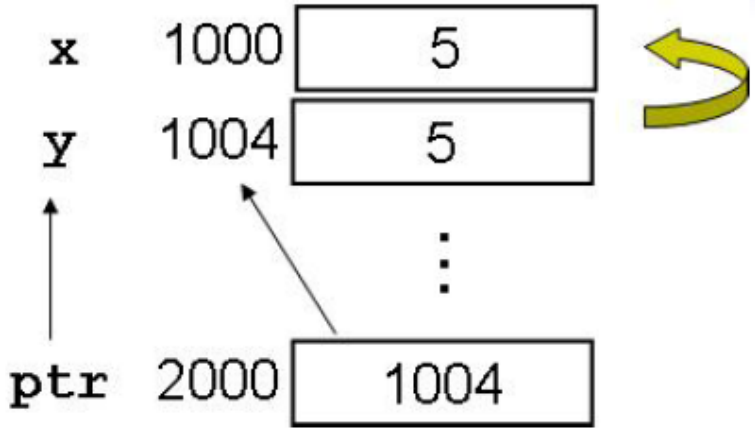
Introduction to C: pointer variables

- Pointer variables are variables that store memory addresses.
- Pointer Declaration:
  - int x, y = 5;
  - int *ptr;
  - /*ptr is a POINTER to an integer variable*/
- Address operator &:
  - ptr = &y;
  - /*assign ptr to the MEMORY ADDRESS of y.*/
- Dereference operator *:
  - x = *ptr;
  - /*assign x to the int that is pointed to by ptr */

Introduction to C: pointer variables

Pointer Example 1

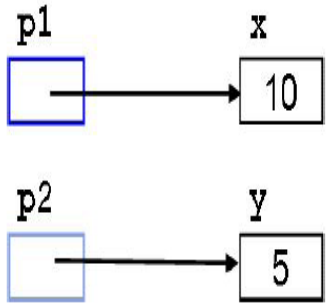
```
int x;
int y = 5;
int *ptr;

ptr = &y;

x = *ptr;
```

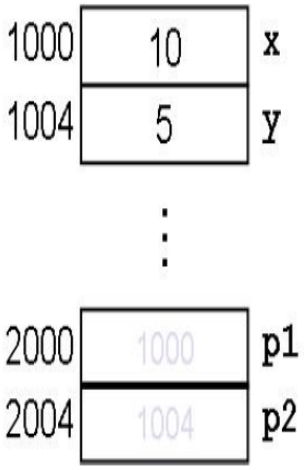


Introduction to C: pointer variables

Pointer Example 2

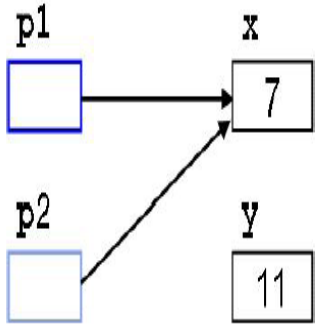
```
int x = 10, y = 5;
int *p1, *p2;
p1 = &x;
p2 = &y;
```

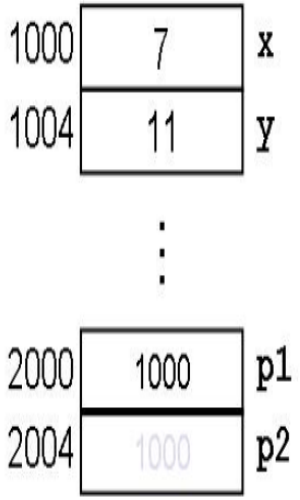




Pointer Example 2

```
p2 = p1;  // Not the same as *p2 = *p1
```





```c
#include <stdio.h>
//swap two values
void swap(int* iPtrX,int* iPtrY);
void fakeswap(int x, int y);
int main()
{
    int x = 10;
    int y = 20;
    int *p1 = &x;
    int *p2 = &y;
    printf("before swap: x=%d y=%d\n",x,y);
    swap(p1,p2);
    printf("after swap: x=%d y=%d\n",x,y);

    printf("----------------------------\n");
    printf("before fakeswap: x=%d y=%d\n",x,y);
    fakeswap(x,y);
    printf("after fakeswap: x=%d y=%d",x,y);
  return 0;
}
```

```c
void swap(int* iPtrX, int* iPtrY)
{
   int temp;
       temp = *iPtrX;
       *iPtrX = *iPtrY;
       *iPtrY = temp;
}
void fakeswap(int x,int y)
{
   int temp;
       temp = x;
       x = y;
       y = temp;
}
```

Introduction to C: Array

Array is a fixed size, sequenced collection of elements of the same data type, with index starts with zero

- Array declaration:
  int a[4];

- Array initialization:
  int a[4] = {3,4,5,6};

- Assignment to and from array element
  a[0] = 3;
  value = a[1];

Introduction to C: Array

Use pointer to access Array

```
int *ptr = a;   // int a[4] in the last slides.
```

Let's set a[1] to 1

```
ptr[1] = 1;
*(ptr+1) = 1;   // we could use ptr+i  to get the address of (i-1)th
                                element in one array
```

```
int i = 3;
ptr2 = a;
ptr = ptr + i; // Ok if i is smaller than the array size
i = ptr - ptr2;  // Ok,  i = 3
ptr = ptr + ptr2;   // Wrong! It's forbidden
```

Introduction to C: struct

```
#include <stdio.h>
//group things together
struct database {
  int id_number;
  int age;
  float salary;
};

int main()
{
  struct database employee;
  employee.age = 22;
  employee.id_number = 1;
  employee.salary = 12000.21;
  printf("Employeee No.%d is %d and his salary is %f\n", employee.id_number,
      employee.age, employee.salary);    // Output:  Employee No.1 is 22 and his salary
      is 12000.21
 return 0;
}
```

```c
#include <stdio.h>

int main()
{
  FILE *ifp, *ofp;
  char *mode = "r";
  char outputFilename[] = "out.list";
  char username[9];
  int score;
  ifp = fopen("in.list", mode);
  if (ifp == NULL) {
    fprintf(stderr, "Can't open input file in.list!\n");
    exit(1);
  }
  ofp = fopen(outputFilename, "w");
  if (ofp == NULL) {
    fprintf(stderr, "Can't open output file %s!\n", outputFilename);
    exit(1);
  }
  while (fscanf(ifp, "%s %d", username, &score) == 2) {
    fprintf(ofp, "%s %d\n", username, score+10);
  }
  fclose(ifp);
  fclose(ofp);
  return 0;
}
```

mode:
r - open for reading
w - open for writing (file need not exist)
a - open for appending (file need not exist)
r+ - open for reading and writing, start at beginning
w+ - open for reading and writing (overwrite file)
a+ - open for reading and writing (append if file exists)

File I/O