Lecture 10 : Pointers in C

Prabhjot Sandhu

TA COMP206
McGill University

October 2, 2018

Recap

Strings and Pointers

C strings are also pointers.

Example

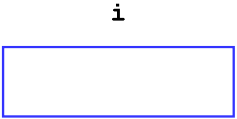
```c
#include<stdio.h>

int main()
{
  char *my_string = "Hello, World!";
  printf("%s\n", my_string);
  return 0;
}
```

Variable

Declaration of a variable

It reserves memory for the variable.

```
int i;
```

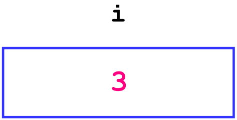


Variable

Initialization of a variable

It assigns a value to the variable along with the declarartion.

```
int i = 3;
```

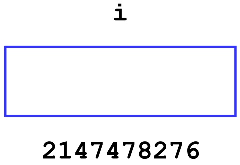


Variable

Declaration of a variable

With the declarartion, the variable gets the address in the memory.

```
int i;
```

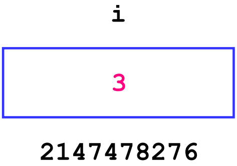


Variable

Address of a variable

It is represented by preceding the variable identifier by an ampersand (&) sign which literally means 'address of'.

```
int i = 3;
```



i

3

2147478276

Address Operator &

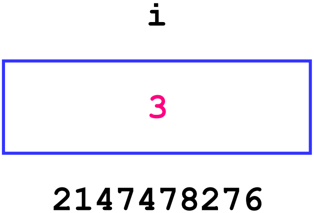
Example

- **Variable identifier** represents *the value of the variable*.
- **Variable identifier preceded with &** represents *the address of the variable*.

```
int i;
i = 3;
printf("The value of i = %d\n", i);
printf("The address of i = %p\n", &i);
```

What is the output?

```
int i = 3;
```



i

3

2147478276

```
printf("The value of i = %d\n", i);
printf("The address of i = %p\n", &i);
```

Pointer

What is a pointer?

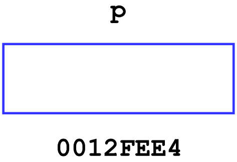
- A pointer is a variable that contains the address of a variable.
- A pointer provides a way of accessing a variable without referring directly to the variable.

Declaration of a pointer variable

```
datatype *pointer_variable;
```

Example

```
int *p;
```



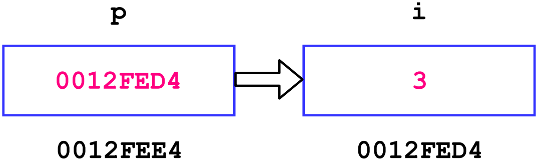
Pointer

Initialization of a pointer

Unlike a simple variable that stores a value, a pointer **must** be initialized with a specified address prior to its use.

Example

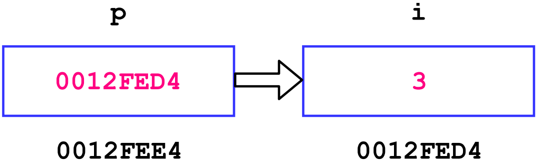
```
int i = 3;
int *p = &i;
```



Pointer

Example

```
int i = 3;
int *p = &i;
printf("The value of i = %d\n", i);
printf("The address of i = %p\n", &i);
printf("The address of i = %p\n", p);
```



p

i

0012FED4

3

0012FEE4

0012FED4

Pointer

Dereferencing

- The primary use of a pointer is to access and change the value of the variable that pointer points to.
- Value of the variable is represented by preceding the pointer variable identifier by an asterisk (*) sign which literally means 'value at address'.
- The 'value at address' operator is also called indirection operator or dereference operator.

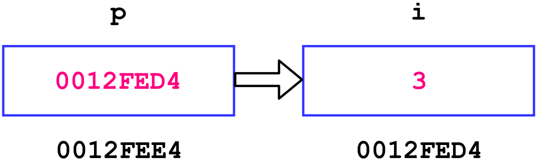
Example

```
int i = 3;
int *p = &i;
printf("The value of i = %d\n", i);
printf("The value of i = %d\n", *p);
```

Pointer

Example

```
int i = 3;
int *p = &i;
printf("The value of i = %d\n", i);
printf("The value of i = %d\n", *p);
printf("The address of i = %p\n", &i);
printf("The address of i = %p\n", p);
printf("The address of p = %p\n", &p);
```



p

i

0012FED4

3

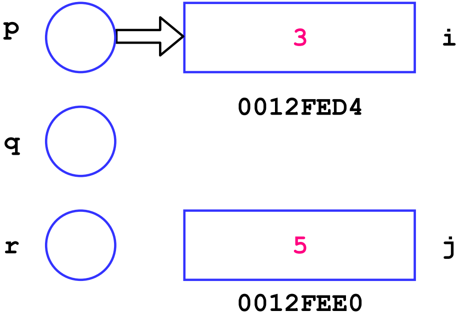
0012FEE4

0012FED4

Pointer

Example

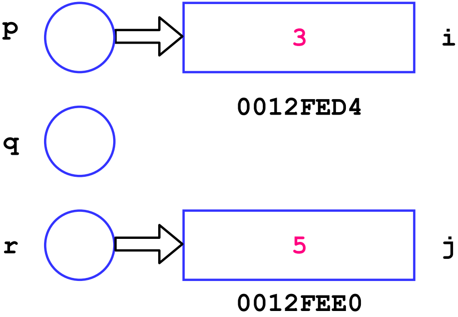
```
int i = 3, j = 5;
int *p = &i;
int *q, *r;
```



Pointer

Example

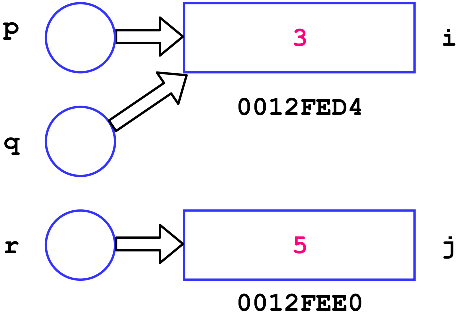
```
int i = 3, j = 5;
int *p = &i;
int *q, *r;
 r = &j;
```



Pointer

Example

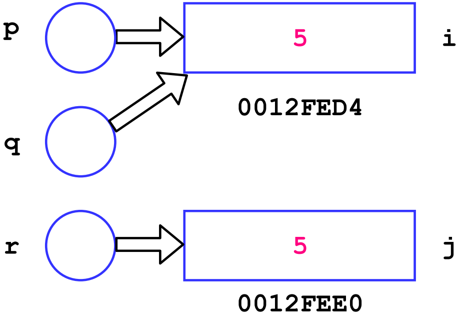
```
int i = 3, j = 5;
int *p = &i;
int *q, *r;
r = &j;
 q = p;
```



Pointer

Example

```
int i = 3, j = 5;
int *p = &i;
int *q, *r;
r = &j;
q = p;
 *p = *r;
```



Void Pointer

- It can point to any data type.
- Pointed data cannot be referenced directly.
- Type casting must be used to turn the void pointer to a concrete data type pointer.

Example

```
int i = 3;
float j = 3.14;
void *p;
p = &i;
printf("The value of i = %d\n", *(int *)(p));
p = &j;
printf("The value of i = %f\n", *(float *)(p));
```

Null Pointer

It is a special pointer value that points nowhere.

Example

```
int *p;
p = NULL;
if(p != NULL)
  printf("%d\n", *p);
```

Use of Pointers

Call by address or pass by reference.

Example – Pass by Value

```
void swap(int a, int b)
{
  int temp;
  temp = a;
  a = b;
  b = temp;
}
int main()
{
  int i=1, j=2;
  printf("%d %d\n", i, j);
  swap(i,j);
  printf("%d %d\n", i, j);
}
```

Use of Pointers

Call by address or pass by reference.

Example – Pass by Reference

```
void swap(int *a, int *b)
{
  int temp;
  temp = *a;
  *a = *b;
  *b = temp;
}
int main()
{
  int i=1, j=2;
  printf("%d %d\n", i, j);
  swap(&i,&j);
  printf("%d %d\n", i, j);
}
```

Use of Pointers

Returning more than one value from a function.

Example

```
float compute(int r, float *p)
{
  float a;
  a = 3.14 * r * r;
  *p = 2 * 3.14 * r;
  return a;
}
int main()
{
  int r = 2;
  float area, perimeter;
  area  = compute(r, &perimeter);
  printf("%f %f\n", area, perimeter);
}
```