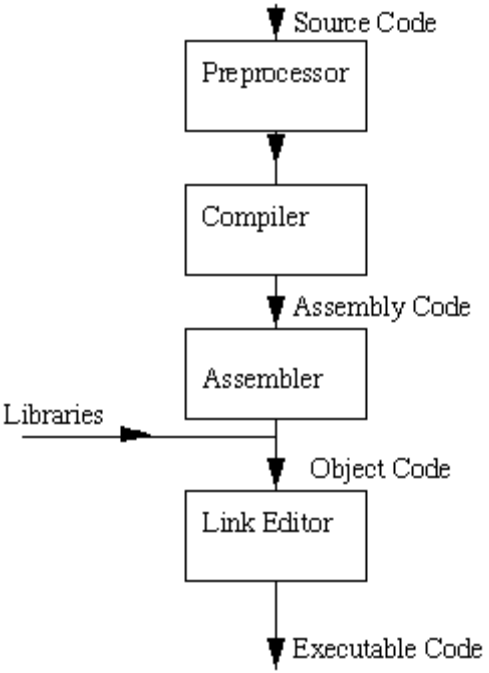
Advanced C in Practice

Prabhat Kumar Padhy

C Program Internals?

- Creating C program, Compiling and Runnings.
  - Create using some editor
  - Compilation → gcc test.c (or) gcc –o test test.c
  - Running → ./test
- C Compilation Model
  - Preprocessor accepts source code as input
  - Responsible for removing comments and interpreting '#'
  - Compiler translates source into assembly code
  - Assembly creates object code (binary)
  - Link editor links the main with other lib/ methods if any
  - Compiler options :
    - -c → gcc –Wall –c file1.c file2.c main.c , gcc file1.o file2.o main.o –o main_ex
    - -library (gcc calc.c –o calc –lm
    - -Ldirectory, -Lpathname, -g , etc …



short number,sum; int bignumber,bigsum; char letter; main() { }

Variable Scoping?

- Constant

```
int testfunc1 (const int a)
{
  return a;
}

int testfunc2 (int const a)
{
  return a;
}
```

- Global Variable

```
short number,sum;
int bignumber,bigsum;
char letter;
int main()
        {
//can do the assignment
here or above as well
        }
```

- Local Variable

```
int main()
{
 int a = 5;
test(a);
printf("Value of
a:%d",a);
}
test (int a)
{
a = 9;
printf("Value of
a:%d",a);
}
```

short number,sum; int bignumber,bigsum; char letter; main() { }

Decision Control?

- If – else

```
if  (expression)
 statement
...or:
  if  (expression)
 statement1
 else
 statement2
...or:
  if  (expression)
 statement1
 else if (expression)
 statement2
 else
statement3
```

- Switch – case

```
switch (letter)
{
 case `A':
 case `E':
 case `I':
 case `O':
 case `U':
        numberofvowels++;
        break;
 case ` ':
        numberofspaces++;
        break;
 default:
        numberofconstants++;
        break;
}
```

- Ternary operator

```
z = (a>b) ? a : b;
```

- Goto ..label

```
main()
{
int a = -1;
if(a<0)
goto label1:
else
printf("value of a:%d",a);
Label1:
printf("value of a:%d",a);
}
```

short number,sum; int bignumber,bigsum; char letter; main() { }

Operator Precedence?

- Operator order
  - Arithmatic operator
  - increment and decrement
  - Comparision operator
  - logical operator
  - Bitwise operator

- Operator order

( ) [ ] -> .

! ~ - * & sizeof cast ++ -

(these are right->left)

* / %

+ -

< <= >= >

== !=

&

^ |

&&

||

?: (right->left)

= += -= (right->left)

, (comma)

- Examples

```
main()
{
int x=3,z;
z=x--- -1;
printf("x=%d z=%d",x,z);
}
main()
{
int i=3,j;
j=++i*++i*++i;
printf("j=%d",j);
}
main()
{
int x,y,z;
x=y=z=-1;
z=++x||++y&&++z;
printf("x=%d y=%d z=%d",x,y,z);
}
```

short number,sum; int bignumber,bigsum; char letter; main() { }

Functions?

```
main()
{
int a,b,c;
a=b=7;
c=mul(a,b);
printf("value of c:%d",c);
}
mul(int p,int q)
{
int r;
r = p*q;
return r;
}
```

```
main()
{
float a,b,c;
float mul();
a=b=1.7;
c=mul(a,b);
printf("value of c:%f",c);
}
float mul(float p,float q)
{
float r;
r = p*q;
return r;
}
```

```
main()
{
float a,b,c;
void mul();
a=b=1.7;
mul(a,b);
}
mul(int p,int q)
{
int r;
r = p*q;
printf("value of r:%f",r);
}
```

short number,sum; int bignumber,bigsum; char letter; main() { }

Functions?

```
//Call by Value
main()
{
int a,b;
a=6;
b=7;
swapv(a,b);
printf("a:%d b:%d",a,b);
}
swap(int p, int q)
{
int r;
r = p;
p=q;
q=r
printf("p:%d q:%d",p,q);
}
```

```
//Call by Reference
main()
{
int a,b;
a=6;
b=7;
swapr(&a,&b);
printf("a:%d b:%d",a,b);
}
swap( p, q)
int *p,*q;
{
int r;
r = *p;
*p=*q;
*q=r
printf("p:%d q:%d",*p,*q);
}
```

Case 1:

Changes made in p and q in swapv() are not reflected back to a and b

Case 2:

Changes make in swapr) using p and q are reflected back in a and b.

Pointers?

```
main()
{
int i=30;
int *j,**k;
j= &i;
k= &j;
printf("Address of i: %d %d %d",&I,j,*k);
printf("Address of j: %d %d %d",&j,k);
printf("Address of k: %d ",&k);
printf("Value of i: %d %d %d %d",i,*(&i),*j,**k);
}
```

short number,sum; int bignumber,bigsum; char letter; main() { }

Pointers?

```
main()
{
int a,*b,**c,***d;
b=&a;
c=&b;
d=&c;
printf("%d %d %d
%d",a,*b,**c,**d);
}
```

```
main()
{
int a=3,*b,*c;
b=&a;
c=b;
*c++ = *b++;
c++;
printf("a=%d b=%d
c=%d",a,b,c);
}
```

Pointers?

```
main()
{
int i=30;
int *j,**k;
j= &i;
k= &j;
printf("Address of i: %d %d %d",&I,j,*k);
printf("Address of j: %d %d %d",&j,k);
printf("Address of k: %d ",&k);
printf("Value of i: %d %d %d %d",i,*(&i),*j,**k);
}
```

Questions?