Computer Programming: Skills & Concepts (INF-1-CP1) Loops

7th October, 2010

Summary of Lecture 7

- Summary of Practical 1
- The `descartes.h` package.
- Example program `square.c`

This Lecture

- The `while` statement.
- The `for` statement.
- `fibonnaci.c`
- `prime.c`
- `scanf` and erroneous input.

`while`

```
while (<condition>) {
  <statement_sequence>;
 }
```

`while` means "repeat until failure" (of the `<condition>`). `<statement-sequence>` must alter some parameters involved in `<condition>`. WHY?

Fibonacci Numbers

0 1

0 + 1 = 1

1 + 1 = 2

1 + 2 = 3

2 + 3 = 5

3 + 5 = 8

5 + 8 = 13

8 + 13 =21

fibonnaci.c

```
int main(void) {

  int n, next, count;
  int previous = 0;       /* Fibonacci -1 */
  int current = 1;   /* Fibonacci 0 */
  ...
  count = 0;

  while (count != n) {

    next = previous + current;    // i.e. 2 = 0 + 1
    previous = current;
    current = next;    // after: 2 + 1
    ++count;

  }
  printf("Fibonacci %d is %d", n, current);
  return EXIT_SUCCESS;
}
```


running fibonnaci.c

```
:  ./a.exe
Calculate which Fibonacci number?  0
Fibonacci 0 is 1

:  ./a.exe
Calculate which Fibonacci number?  1
Fibonacci 1 is 1

:  ./a.exe
Calculate which Fibonacci number?  2
Fibonacci 2 is 2

:  ./a.exe
Calculate which Fibonacci number?  7
Fibonacci 7 is 21
```


while-statement: Repeat n-times

```
initialise_iterator;
  while (<not_iterator_endpoint>) {
    <statement_sequence>;

    next_iterator_value;
 }
```

while-statement

Counting-up:

```
count = 0;
while (count != n) {
  <statement_sequence>;
  ++count;
}
```

Counting-down:

```
count = n;
  while (count != 0 ) {
    <statement_sequence>;
    --count;
  }
```


The for-loop

```
for (count = n; count != 0; --count) {
  <statement_sequence>;
 }
```


Fibonacci using for

```
int n, next, count;
int previous = 0; /* Fibonacci -1 */
int current = 1; /* Fibonacci 0*/
for (count = n; count != 0; --count) {
  next = previous + current;
  previous = current;
  current = next;
}
```


Prime Numbers

Definition: A prime number is any natural number which has no factors except itself and 1.

Prime: 3, 7, 11

Not Prime: 9 (3*3), 10 (2*5)
Simple test for primes:

`n` is prime if `n=1` or if there is no integer `k` between 2 and `sqrt(n)` such that `n % k = 0`.

prime.c

```
...
k = 2; // First divisor-attempted is 2
int prime = 1;
while (((k* k) <= n) && (prime)) { // finish at sqrt(n)
  if ((n % k) == 0) {
    printf(("%d is %d * %d\n", n, n/k, k);
    prime = 0; // terminate the loop
   }
  ++k; // Test each value
 }
if (prime)
  printf("%d is a prime number\n", n);
return EXIT_SUCCESS;
```