Optimization examples

```
for(i=0; i<10000; i++);
```

← delay

optimizing compiler — remove this code

```
for(i=0; i<10000; i++) {
    a[i] = b[i];
}
```

← this has useful work

```
int function_mine (int z) {
  int a[10000], b[10000];
  .
  .
  .
  return (z+1);
}
```

Assignment — write the function int c_to_f (int);

ECGR 5101 / 4101 11/4/14

Test Case: CtoF001
Objective: Test function C_to_f
Passing: Expected value returned
Materials needed: None

1. Run function, input -32768, expect -32768 to be returned (number too low error)
2. Run function, input 32767, expect -32768 (or -32768) to be returned (number too high error)

Test Case: CtoF002

1. Run function, input -274, expect -32768 to be returned (number too low error
2. Run function, input -273, expect -459 to be returned (correct usage)
3. Run function, input -272, expect ____ to be returned (correct usage)

Test harness

```
int testnum[12];

testnum[0] = C_to_F(-32768);
testnum[1] = C_to_F(-274);
testnum[2] = C_to_F(-273);
   .
   .
   .
```

Then print, or examine testnum array

Andrew Maldonado
William Myers
Leonard O'Shields

10/10

```
#define HIGHEST_C (32768)
#define FUNCT_ER (-32768)
#define C_FREEZE (32.0)
#define C_TO_F_CONST (9.0/5.0)

int C_to_f(int C) {
    int val;
    if ( C > HIGHEST_C )
        return (FUNCT_ER);
    val = (int)((float)(C + C_FREEZE) * C_TO_F_CONST));
    return (val);
}
```

if needs to be added checks below -273 is Lowest C

errors, but since we did a code review, I gave them full marks

32767°F

Lou Deguzman
Tom Perkins

10/10

```
#define   HIGHEST          (18186)
#define   LOWEST           (-273)
#define   FUNC_ERR_LOW     (-32768)
#define   FUNC_ERR_HI      (-32767)
#define   C2F_CONST        (9.0/5.0)
#define   F2C_OFF          (32.0)

int  c_to_f (int c) {

     if (c < LOWEST)   return (FUNC_ERR_LOW);
     if (c >= HIGHEST) return (FUNC_ERR_HI);
     return (int)((float)c * C2F_CONST + F2C_OFF);

}
```

don't match

errors, but since we did a code review, they get full marks

Quiz 13

Khawaja Amin
Brian Mason

```
#define LOWEST_C      (-273)
#define FUNCT_ERROR   (-32768)
#define F_FREEZE      (32.0)
#define C_TO_F_CONST  (9.0/5.0)
#define ROUND         (0.5)
#define HIGHEST_C     (32768.0/C_TO_F_CONST)

int C_to_F( int c) {
    if (c < LOWEST_C || c > HIGHEST_C)
        return FUNCT_ERROR;
    float val;
    val = ((float)c * C_TO_F_CONST) + F_FREEZE;
    if (val < 0.0)
        return (int) (val - ROUND);
    else
        return (int) (val + ROUND);
}
```

-273

10/10 guaranteed since I used it as an example

32767

can use it?

Subtract 32 from this

Verify what the compiler asm will do

ESHWARA PRASATH RAMESH
800865777
BALAJI MASANAMUTHU CHINNATHURAI
800876530

10/10 guaranteed since I used it as an example

```
#define LOWEST_C  (-273)
#define FUNCT_ERROR (-32768)
#define FREEZE (32.0)
#define C_TO_F_CONST (9.0/5.0)

int c_to_f (int c)
{
  int val;
  if (c < LOWEST_C) return (FUNCT_ERROR);
  val = (int) (((float) c * C_TO_F_CONST) + FREEZE);
  return (val);
}
```

Overflow GT 32767

round correctly for negative numbers