Part 1

1- All types that are not structures or enums:

- typedef int32_t CnetAddr;
- typedef char* CnetColour;
- typedef intptr_t CnetData;
- typedef void* CnetRandom;
- typedef int64_t CnetTime;
- typedef int32_t CnetTimerID;
- typedef unsigned char CnetNICAddr[LEN_NICADDR];

Of the above, the types that not equivalent to 'int' are:

- CnetColour
- CnetData
- CnetRandom
- CnetTime
- CnetNICAddr

2- The node's current simulation time is defined by the CnetNodeInfo structure. It can be accessed through the nodeinfo.time_in_usec field. Printing with printf can be done using one of the following methods:

```
long time_value = nodeinfo.time_in_usec;
printf( "time = %lld usecs \n", time_value );    // option 1
printf( "time = %s usec \n", CNET_format64(time_value) );//option 2
```

3- `cnet.h` does not specify the structure of application layer messages.

4- When the CHECK macro receives a non-zero return value from an enclosed function, it calls CNET_exit() and pops up a GUI window showing the file, calling function, line number, and an error code value for the error.

We can use a modified version of the CHECK macro (e.g. CHECK_C presented in the more practice for lab 1) to check and print errors without exiting the simulation.

```
#define CHECK_C(call)                    if ((call) != 0) LAB_warn (FILE, func,_LINE);

void LAB_warn(const char *filenm, const char *function, int lineno)
{
    char            msg[240];
    sprintf(msg, "Warning: %s (%s usec, '%s:%d'): %s", nodeinfo.nodename,
                CNET_format64(nodeinfo.time_in_usec), function, lineno, cnet_errname[cnet_errno]);
    fprintf(stderr,"%s\n",msg);
 }
```

5- FRAME_SIZE(f) and sizeof(f) can produce different values. Note that FRAME_SIZE(f) provides the size of the frame header plus the message length. However, sizeof(f) is the frame header size plus the maximum message size as the message char array is pre-allocated with a max size that may not be fully used. In short, FRAME_SIZE(f) gives the actual size of the frame in use, whereas sizeof(f) gives the maximum size of the frame.

6- The protocol in stopandwait.c uses a checksum to detect corrupted frames. The sender computes the checksum using the following line:

```
f.checksum = CNET_ccitt((unsigned char*) &f, (int) length);
```

Note that the checksum is stored and transmitted in the frame. Upon receipt, the checksum is recomputed in the same way and compared against the transmitted checksum in the frame. Any mismatch indicates a corrupted frame.

```
if( CNET_ccitt ((unsigned char *)&f,  (int)len) != checksum )
    {  printf("\t\t\t\tBAD checksum - frame ignored\n");
    return;        // bad checksum, ignore frame
  }
```

7- When an ACK is received, the CNET_stop_timer () function is called to stop the timer that is initialized upon transmission of the data frame.
