We have to modify the function *get_ifi_info* so that it also gets the network mask for the *IP* address it locates.

The complete modified code is available for you to copy, from directory ***~cse533/Asgn2_code***. Make sure you read the *README* file there to understand how function and file names have been changed, typically by adding '*_plus*' to their original name. For example, the modified *get_ifi_info* function is called *get_ifi_info_plus*, and so on.

---

get_ifi_info builds a linked list of *ifi_info* structures in which it copies the information it obtains. The first step then is to add an extra field to the *ifi_info* structure, in which to store the network mask. The *ifi_info* structure is defined in file *ioctl/unpifi.h* (Figure 17.5, *p*.471). The modifications needed are shown below. (The line numbers shown refer to Figure 17.5 so that you can see exactly where the new code needs to be introduced.).

```
 9 struct ifi_info {
10     char    ifi_name[IFI_NAME]; /* interface name, null terminated */
11     short   ifi_index;                      /* interface index */
12     short   ifi_mtu;                        /* interface MTU */
13     u_char  ifi_haddr[IFI_HADDR];   /* hardware address */
14     u_short ifi_hlen;           /* #bytes in hardware address: 0, 6, 8 */
15     short   ifi_flags;          /* IFF_xxx constants from <net/if.h> */
16     short   ifi_myflags;        /* our own IFI_xxx flags */
17     struct sockaddr  *ifi_addr; /* primary address */
18     struct sockaddr  *ifi_brdaddr;  /* broadcast address */
19     struct sockaddr  *ifi_dstaddr;  /* destination address */

/*================ cse 533 Assignment 2 modifications =================*/

       struct sockaddr  *ifi_ntmaddr;  /* netmask address */

/*=====================================================================*/

20     struct ifi_info  *ifi_next;      /* next of these structures */
21 };
```

The next step is to modify *get_ifi_info* itself so that it calls *ioctl* to obtain the network mask and put it in the *ifi_info* structure. The changes are shown below (*cf.* Figure 17.10, *p*.479).

```
117                     ifi->ifi_dstaddr = Calloc(1, sizeof(struct sockaddr_in));
118                     memcpy(ifi->ifi_dstaddr, sinptr, sizeof(struct sockaddr_in));
119                 }
120 #endif

/*================== cse 533  Assignment 2 modifications ==================*/

   #ifdef  SIOCGIFNETMASK
                    Ioctl(sockfd, SIOCGIFNETMASK, &ifrcopy);
                    sinptr = (struct sockaddr_in *) &ifrcopy.ifr_addr;
                    ifi->ifi_ntmaddr = Calloc(1, sizeof(struct sockaddr_in));
                    memcpy(ifi->ifi_ntmaddr, sinptr, sizeof(struct sockaddr_in));
   #endif

/*=====================================================================*/

121           break;
```

To test that our modifications work, we use the *prifinfo* program to print out the information gathered by *get_ifi_info* in the linked list of *ifi_info* structures. But first, we must modifiy *prifinfo* (*cf*. Figure 17.6, *p*.472) so that it also prints out the new network mask field we introduced into the *ifi_info* structure above.

```
39              if ( (sa = ifi->ifi_addr) != NULL)
40                      printf("  IP addr: %s\n", Sock_ntop_host(sa, sizeof(*sa)));

/*=================== cse 533 Assignment 2 modifications ======================*/

                if ( (sa = ifi->ifi_ntmaddr) != NULL)
                        printf("  network mask: %s\n",
                                                Sock_ntop_host(sa, sizeof(*sa)));

/*=============================================================================*/

41              if ( (sa = ifi->ifi_brdaddr) != NULL)
42                      printf("  broadcast addr: %s\n",
43                                              Sock_ntop_host(sa, sizeof(*sa)));
```

Finally, *prifinfo* calls function *free_ifi_info* which must also be slightly modified (*cf*. Figure 17.11, *p*.480) to ensure it also frees the socket address structures, pointed to from the structures *ifi_info*, in which the network masks are stored.

```
152             if (ifi->ifi_dstaddr != NULL)
153                     free(ifi->ifi_dstaddr);

/*============ cse 533 Assignment 2 modifications =============*/

                if (ifi->ifi_ntmaddr != NULL)
                        free(ifi->ifi_ntmaddr);

/*=============================================================*/

154             ifinext = ifi->ifi_next;   /* can't fetch ifi_next after free() */
155             free(ifi);
```