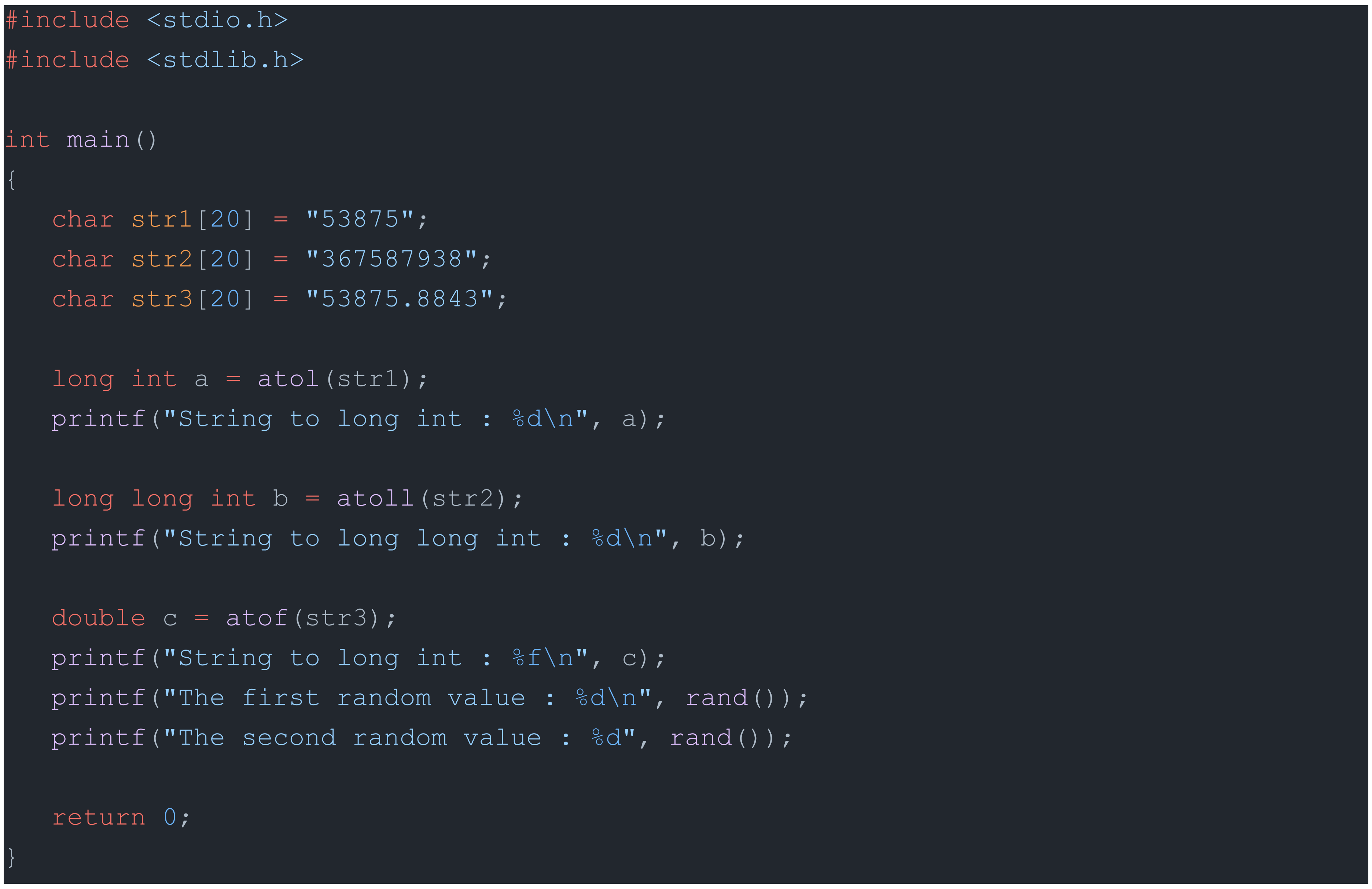
Gita Alekhya Paul **20/07/2021**

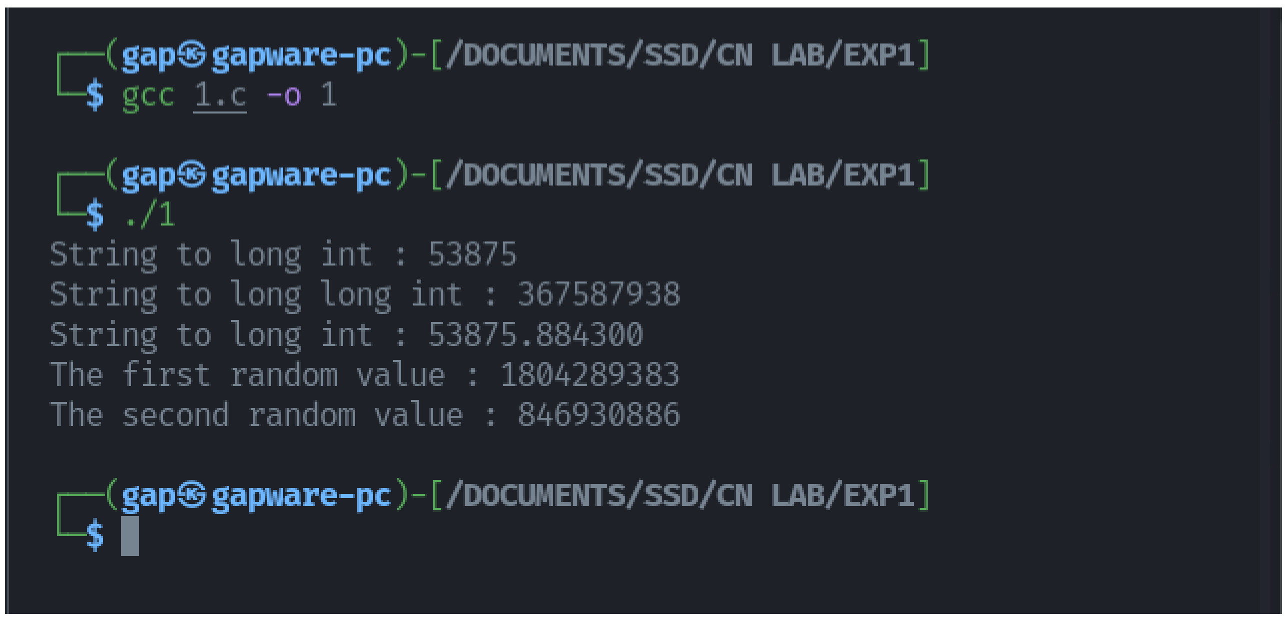
Computer Networks
RA1911030010014
Experiment - 1

Aim: **Study of Header Files with Respect to Socket Programming**

1. **<stdio.h> and <stdlib.h>**



```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
   char str1[20] = "53875";
   char str2[20] = "367587938";
   char str3[20] = "53875.8843";

   long int a = atol(str1);
   printf("String to long int : %d\n", a);

   long long int b = atoll(str2);
   printf("String to long long int : %d\n", b);

   double c = atof(str3);
   printf("String to long int : %f\n", c);
   printf("The first random value : %d\n", rand());
   printf("The second random value : %d", rand());

   return 0;
}
```

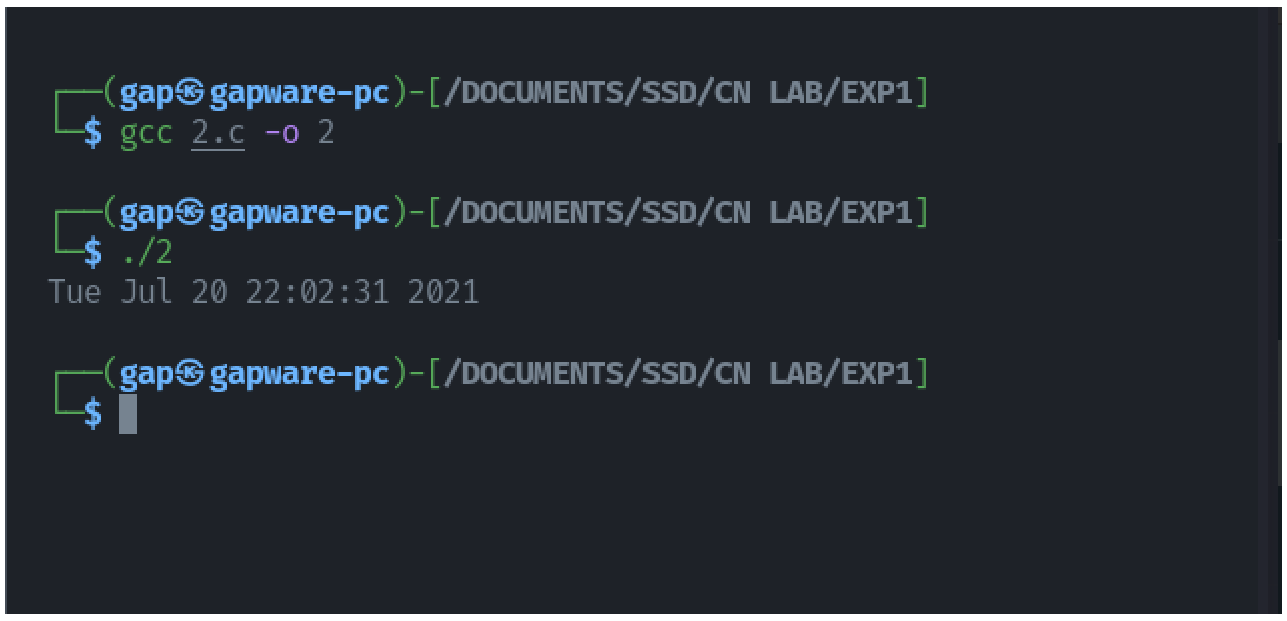


```
┌──(gap㉿gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP1]
└─$ gcc 1.c -o 1

┌──(gap㉿gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP1]
└─$ ./1
String to long int : 53875
String to long long int : 367587938
String to long int : 53875.884300
The first random value : 1804289383
The second random value : 846930886

┌──(gap㉿gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP1]
└─$ 
```

Gita Alekhya Paul **20/07/2021**

2. **<time.h>**

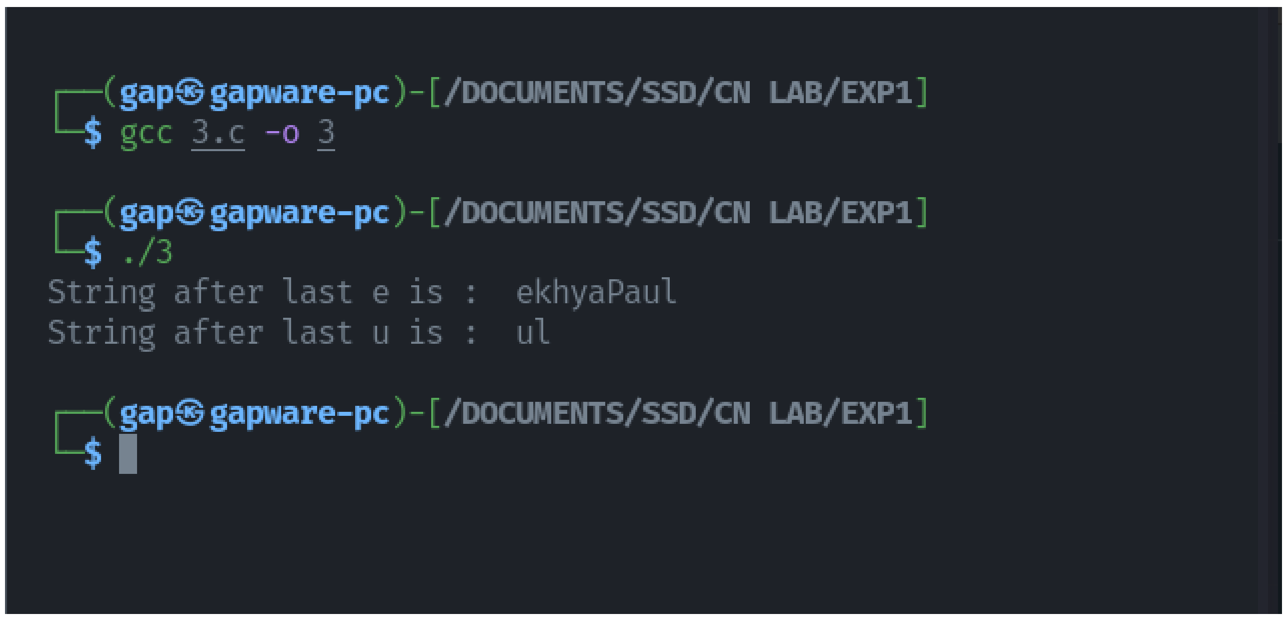
```c
#include <stdio.h>
#include <time.h>
int main(void)
{
   struct tm *ptr;
   time_t t;
   t = time(NULL);
   ptr = localtime(&t);
   printf("%s", asctime(ptr));
   return 0;
}
```



```
┌──(gap㉿gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP1]
└─$ gcc 2.c -o 2

┌──(gap㉿gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP1]
└─$ ./2
Tue Jul 20 22:02:31 2021

┌──(gap㉿gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP1]
└─$ 
```

3. **<string.h>**

```c
#include <stdio.h>
#include <string.h>

int main()
{

   char st[] = "GitaAlekhyaPaul";
   char ch = 'e';
   char *val;

   val = strrchr(st, ch);

   printf("String after last %c is :  %s \n",
          ch, val);

   char ch2 = 'u';
   val = strrchr(st, ch2);

   printf("String after last %c is :  %s ",
          ch2, val);

   return (0);
}
```

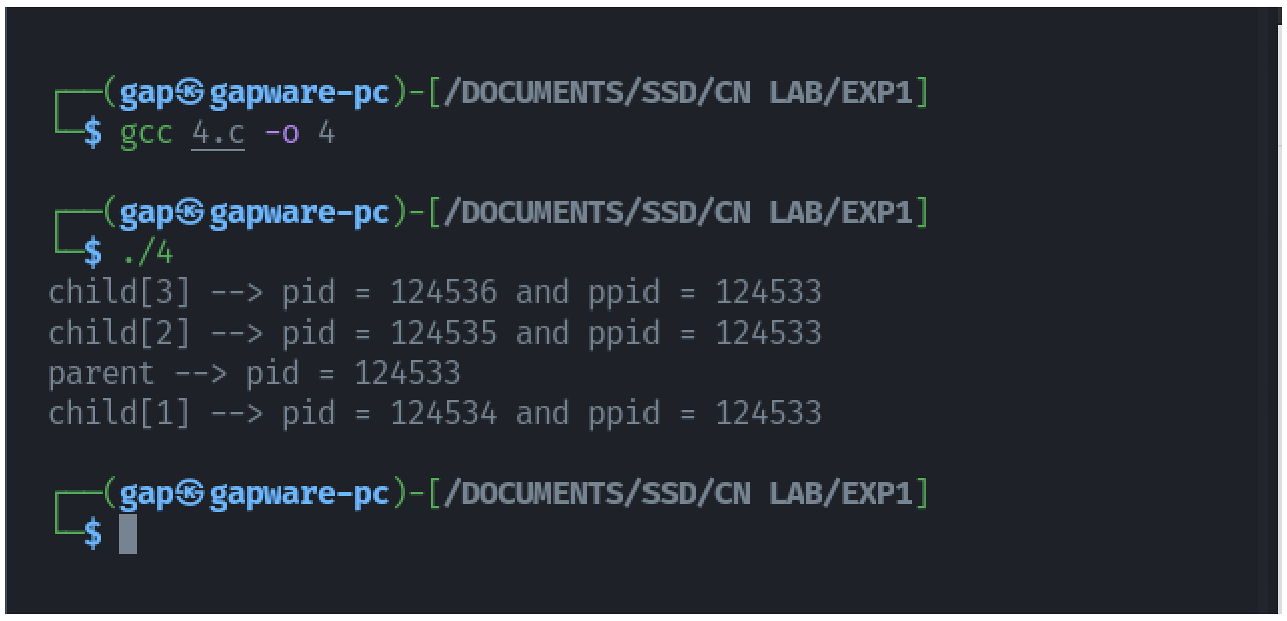


```
┌──(gap㉿gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP1]
└─$ gcc 3.c -o 3

┌──(gap㉿gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP1]
└─$ ./3
String after last e is :  ekhyaPaul
String after last u is :  ul

┌──(gap㉿gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP1]
└─$ 
```

4. **<unistd.h>**

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main()
{
   int pid, pid1, pid2;

   pid = fork();

   if (pid == 0)
   {

       sleep(3);

       printf("child[1] --> pid = %d and ppid = %d\n",
              getpid(), getppid());
   }

   else
   {
       pid1 = fork();
       if (pid1 == 0)
       {
           sleep(2);
           printf("child[2] --> pid = %d and ppid = %d\n",
                  getpid(), getppid());
       }
       else
       {
           pid2 = fork();
           if (pid2 == 0)
           {
               printf("child[3] --> pid = %d and ppid = %d\n",
                      getpid(), getppid());
```

```
            }

            else
            {
                sleep(3);
                printf("parent --> pid = %d\n", getpid());
            }
        }
    }

    return 0;
}
```



```
┌──(gap㉿gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP1]
└─$ gcc 4.c -o 4

┌──(gap㉿gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP1]
└─$ ./4
child[3] --> pid = 124536 and ppid = 124533
child[2] --> pid = 124535 and ppid = 124533
parent --> pid = 124533
child[1] --> pid = 124534 and ppid = 124533

┌──(gap㉿gapware-pc)-[/DOCUMENTS/SSD/CN LAB/EXP1]
└─$ 
```

5. **<sys/types.h>**
   Defines the data type of socket address structure in unsigned long.
   The <sys/types.h> header shall include definitions for at least the following types:
   a. Blkcnt_t
      Used for file block counts.
   b. Blksize_t
      Used for block sizes.
   c. Clock_t
      Used for system times in clock ticks or CLOCKS_PER_SEC;
   d. Clockid_t
      Used for clock ID type in the clock and timer functions.
   e. Dev_t
      Used for device IDs.

6. **<sys/socket.h>**
   The socket functions can be defined as taking pointers to the generic socket address structure called **sockaddr**.
   The <sys/socket.h> header shall define the type **socklen_t,** which is an integer type of width of at least 32 bits; see APPLICATION USAGE.
   The <sys/socket.h> header shall define the unsigned integer type **sa_family_t**.
   The <sys/socket.h> header shall define the sockaddr structure that includes at least the following members:
   a. sa_family_t sa_family Address family.
   b. char sa_data[] Socket address (variable-length data)

7. **<netinet/in.h>**
   Defines the IPv4 socket address structure commonly called Internet socket address structure called sockaddr_in.
   The <netinet/in.h> header shall define the following types:
   in_port_t
   Equivalent to the type **uint16_t** as defined in <inttypes.h> .
   in_addr_t
   Equivalent to the type **uint32_t** as defined in <inttypes.h> .
   The **sa_family_t** type shall be defined as described in <sys/socket.h>.

The **uint8_t** and **uint32_t** type shall be defined as described in <inttypes.h>. Inclusion of the <netinet/in.h> header may also make visible all symbols from <inttypes.h> and <sys/socket.h>.

8. **<netdb.h>**
   Defines the structure hostent for using the system call gethostbyname to get the network host entry.
   The <netdb.h> header may make available the type in_port_t and the type in_addr_t as defined in the description of <netinet/in.h>.
9. **<sys/stat.h>**
   Contains the structure stat to test a descriptor to see if it is of a specified type. Also it is used to display file or file system status.stat() updates any time related fields.when copying from 1 file to another.
10. **<sys/ioctl.h>**
    Macros and defines used in specifying an ioctl request are located in this header file. We use the function ioctl() that is defined in this header file. ioctl() function is used to perform ARP cache operations.
11. **<pcap.h>**
    Has function definitions that are required for packet capturing. Some of the functions are pcap_lookupdev(),pcap_open_live() and pcap_loop(). pcap_lookupdev() is used to initialize the network device.The device to be sniffed is opened using the pcap_open_live(). Pcap_loop() determines the number of packets to be sniffed.
12. **<net/if_arp.h>**
    Contains the definitions for Address Resolution Protocol. We use this to manipulate the ARP request structure and its data members arp_pa,arp_dev and arp_ha. The arp_ha structure's data member sa_data[ ] has the hardware address.
13. **<errno.h>**
    It sets an error number when an error and that error can be displayed using perror function. It has symbolic error names. The error number is never set to zero by any library function.
14. **<arpa/inet.h>**
    This is used to convert internet addresses between ASCII strings and network byte ordered binary values (values that are stored in socket address structures). It is used for inet_aton, inet_addr, inet_ntoa functions.