Write a C program to print the address of a variable and enter a long loop (say using while(1)).

a) Start three to four processes of the same program and observe the printed address values.


```c
GNU nano 6.2                                    exp10_a.c
#include<stdio.h>
#include<sys/types.h>
#include<unistd.h>

int main()
{
fork();
fork();

int var=1,i=1;

while(1)
{
if(i==5)
{
break;
}
printf("Address of var in loop =%p\n",&var);
i++;
}
return 0;
}
```
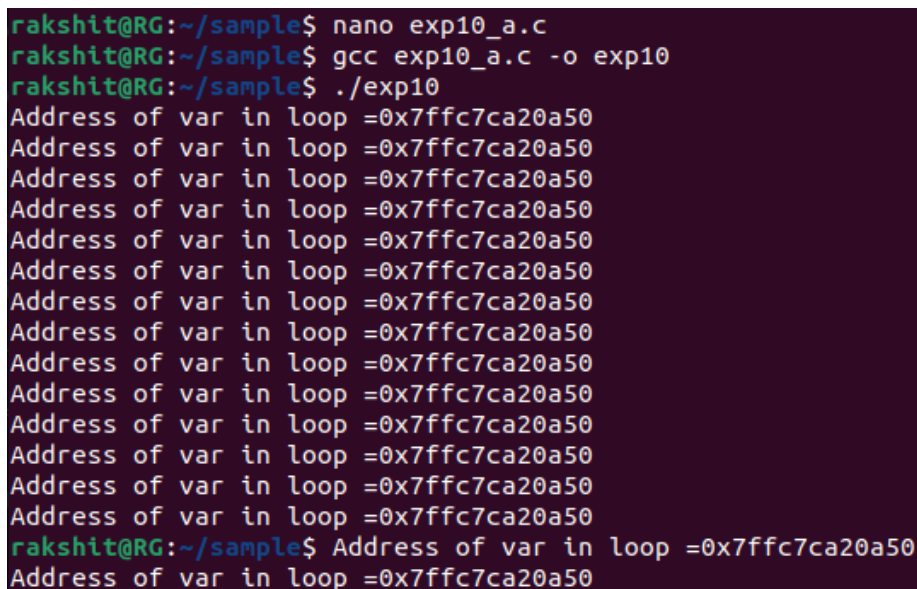
```
rakshit@RG:~/sample$ nano exp10_a.c
rakshit@RG:~/sample$ gcc exp10_a.c -o exp10
rakshit@RG:~/sample$ ./exp10
Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
rakshit@RG:~/sample$ Address of var in loop =0x7ffc7ca20a50
Address of var in loop =0x7ffc7ca20a50
```

b) Show how two processes which are members of the relationship parentchild are concurrent from execution point of view, initially the child is copy of the parent, but every process has its own data.

```c
  GNU nano 6.2                                    exp10_b.c
#include <unistd.h>
#include <sys/types.h>
#include <errno.h>
#include <stdio.h>
#include <sys/wait.h>
#include <stdlib.h>

int main(void)

{ //declare variable

int var=1;

int* p = (int*) malloc(2);
 pid_t PID = fork();
*p = 0;
if (PID >= 0)
{
if (PID ==0)
{

printf("\n\nChild Process:\nInitial Value = %d", var);
var=5;
printf("\nNew Value of var = %d", var);
printf("\nAddress of malloc in child= %p", p);
 printf("\nAddress of var in child= %p\n", &var);
}
else
{
printf("\n\nParent process:\nInitial Value = %d", var);
var = 10;
printf("\nNew Value = %d", var);
printf("\nAddress of malloc in parent= %p",p);
printf("\nAddress of var in child= %p\n", &var);
}
}
return 0;
}
```

```
rakshit@RG:~/sample$ ./exp10


Parent process:
Initial Value = 1
New Value = 10
Address of malloc in parent= 0x55be6a01c2a0
Address of var in child= 0x7ffc6dc3eaf8


Child Process:
Initial Value = 1
New Value of var = 5
Address of malloc in child= 0x55be6a01c2a0
Address of var in child= 0x7ffc6dc3eaf8
rakshit@RG:~/sample$
```