Linux System Programming Assignment 13

1.Write a program which creates one shared object library which contains two funcntions named as search() and sort().

Search function will accept one array and innteger and check whether that innteger is available in that array or not.

Sort function will accept the array and sort the contents of that array using any technique. Use all the functions of that shared object by writing the client application

----

```
//header.h

#define True 1
#define False 0

typedef int BOOL;

void sort(int iarr[], int isize);
int search(int iarr[], int isize, int icmp);
int add(int no1, int no2);
```

```
//sort.c
void sort(int iarr[], int isize)
{
        int i =0, j = 0, itemp = 0;

        for(i = 0;i < isize; i++)
        {
                for(j = i + 1; j < isize; j++)
                {
                        if(iarr[i] > iarr[j])
                        {
                                itemp = iarr[j];                        //swappig on address
                                iarr[j] = iarr[i];
                                iarr[i] = itemp;
                        }
                }
        }
}
```

```
//search.c

int search(int iarr[], int isize,  int icmp)
{
        int i= 0;
```

```
	for(i = 0; i < isize; i++)
	{
		if(iarr[i] == icmp)
		{
			break;
		}
	}

	if(i == isize)
	{
		return 0;
	}
	else
	{
		return 1;
	}
}
```
----

creating .c to .o files
$gcc -g -c -fPIC search.c sort.c

now sort.o and search.o new files get created

next .o to so creating
$gcc -g -shared -o libfuns.so sort.o search.o

now libfuns.so file gets created

this file .so file we open in our main function

last: creating executable
$ gcc main.c -o myexe -ldl

myexe file gets created to execute , we can use to on shell to see interface of our app

```
//main.c



#include<stdio.h>
#include<unistd.h>
#include<dlfcn.h>
#include<stdlib.h>

typedef int BOOL;

int main()
{
	void *lib = NULL;
	BOOL result = 0;
```

```
//create function pointer
int(*iptr)(int*,int,int);           //searching
void(*vptr)(int*,int);              //sorting
int(*padd)(int,int);                //addition

int arr[] = {12,4,1,41,5,19};
int ival = 5, i = 0;

lib = dlopen("./libfuns.so", RTLD_LAZY);   //opening so file and getting its address
if(lib == NULL)
{
        printf("Unable to open so file\n");
        exit(-1);
}

printf("shared object file opened successfully\n");


/////////////////////////
//    for searching
//////////////////////

iptr = dlsym(lib,"search");                         //getting targeted function address

if(iptr == NULL)
{
        printf("Unable to get address of search fuction \n");
        exit(-1);
}
printf("address of search got successfully\n");


result  = iptr(arr, sizeof(arr)/sizeof(arr[0]), ival);                 //call to function using address

if(result == 0)
{
        printf("%d does not exist into given array\n",ival);
        exit(-1);
}
else
{
        printf("%d exist into given array\n",ival);
}
```

```
	///////////////////////////////
	//	for sorting
	//////////////////////////////

	vptr = dlsym(lib, "sort");			//getting address of sort fuction

	if(vptr == NULL)
	{
		printf("Unable to get sort function address\n");
		exit(-1);
	}

	printf("Sort function address got\n");

	printf("before sort\n");
	for(i = 0; i < sizeof(arr)/sizeof(arr[0]);i++)
	{
		printf("%d ",arr[i]);
	}

	vptr(arr, sizeof(arr)/sizeof(int));			//calling to sort function using address

	printf("\nafter sort\n");
	for(i = 0; i < sizeof(arr)/sizeof(arr[0]);i++)
	{
		printf("%d ",arr[i]);
	}
	printf("\n");


	dlclose(lib);

	exit(EXIT_SUCCESS);
}
```

/*
2. Write a program which creates one shared object file which contains one function named as SumFactors(). SumFactor function accept one integer and return the summation of all its factors.

Use the function of that sharaed object by writing the client application.

input: 40
1 + 2 + 4 + 5 + 8 + 10 + 20 = 50
***/**

```
//main.c

#include<stdio.h>
#include<stdlib.h>
#include<dlfcn.h>
#include "headers.h"

int main()
{

	int ival = 40, result = 0;
	int(*fptr)(int);		//function poitner prototype
	void *lib = NULL;



	lib = dlopen("./myshared.so", RTLD_LAZY);
	if(lib == NULL)
	{
		printf("Error to opon SumFacts.so file\n");
		exit(-1);
	}
	printf("shared object file opened successfully\n");




	fptr = dlsym(lib, "sumFactors");
	if(fptr == NULL)
	{
		printf("Error to get address of function\n");
		exit(-1);
	}
	printf("sumFactors function's got address\n");


	result = fptr(ival);
	printf("Sum of %d's factors is: %d\n",ival, result);

	dlclose(lib);


	exit(EXIT_SUCCESS);
}
```

```
//headers.h

int sumFactors(int iNo);
```

```
//helper.c

int sumFactors(int iNo)
{
	int isum = 0;
	int i = 1;

	while(i <= iNo/2)
	{
		if(iNo  % i == 0)
		{
			isum += i;
		}

		i++;
	}

	return isum;
}
```

```
$gcc -g -c -fPIC helper.c
```

it created helper.o file

now created .so using .o file

```
$gcc -g  -shared -o helper.so helper.o
```

now we use this .so into our main function

/*
3. Write a program which creates one shared object file which contains one function named as CheckPerfect(). CheckPerfect function internally loads the shared object which was created
in the above question and call the function SumFactors to check whether the number is perfect or not.
Use the function of that sharaed object by writing the client application.

***/**

```
//main.c

#include<stdio.h>
#include<stdlib.h>
#include<dlfcn.h>

#include"headers.h"

int main()
```

```
{
	int ival = 8128;
	BOOL result = 0;

	void *lib = NULL;
	int(*fptr)(int);

	lib = dlopen("./helpers.so", RTLD_LAZY);
	if(lib == NULL)
	{
		printf("Unable to open sumFacts.so file\n");
		exit(-1);
	}
	printf(".so opened\n");



	fptr = dlsym(lib, "checkPerfect");
	if(fptr == NULL)
	{
		printf("Unable to get address sumfactors function\n");
		exit(-1);
	}
	printf("function address got\n");




	result = fptr(ival);
	if(result)
	{
		printf("Given Num %d is perfect\n", ival);
	}
	else
	{
		printf("Given Num %d is not perfect\n", ival);
	}


	exit(EXIT_SUCCESS);
}

 -----------------------------------------------------------------------------------------------
//helpers.c
//def of checkperfect
//def of sumfactors

#include<stdio.h>


int sumfactors(int iNo)
{
	printf("inside sunfactors\n");
	int i = 1, isum = 0;

	while(i <= iNo/2)
	{

		if(iNo % i == 0)
		{
				isum += i;
		}
		i++;
	}

	return isum;
}

int checkPerfect(int iNo)
{
	if(iNo == (sumfactors(iNo)))
	{
		return 1;
	}
	else
	{
		return 0;
	}

}
```

```
//headers.h
#include<stdio.h>
#include<dlfcn.h>
#include<stdlib.h>

typedef int BOOL;

#define True 1
#define False 0

int sumfactors(int);
BOOL checkPerfect(int);
```

----

to create .so of helpers.c cmd is

```
$gcc -g -c -fPIC helpers.c
```

it will generate helpers.o file

now .o to so conversion cmd

```
$gcc -g -shared -o helpers.so helpers.o
```

it creates .so file which contains our function definition, we open this .so file gets address of function into pointer.
//fucntions use on .so file
dlopen("file.so path", flags); return address of loaded file into memory
dlsym(address of memory, "functionname"); //return address of targeted function
dlclose(address of memory); //closing .so file

/*
4. Write a program which creates two separate shared object files one contains the the function named as MAX() other file contains the funcntion named as MIN().
MAX function accept array of integer and return the maximum element. MIN function accept array from user and return the maximum element.

Load both the shared object files from one client application and call both the functions.
***/**

```
//main.c
#include<stdio.h>
#include"headers.h"
#include<stdlib.h>
#include<dlfcn.h>


int main()
{

	int res = 0;
	int arr[] = {121,111,424,211,324}, result = 0;
	char ch = ',';

	void *lib1 = NULL, *lib2 = NULL;

	//function pointer prototype
	int(*fptr1)(int*, int);
	int(*fptr2)(int*, int);


	lib1 = dlopen("./Max.so", RTLD_LAZY);

	if(lib1 == NULL)
	{
		printf("Unable to open Max.so\n");
		exit(-1);
	}


	fptr1 = dlsym(lib1, "_max");
	if(fptr1 == NULL)
	{
		printf("Unable to get address of max function\n");
		exit(-1);
	}

	printf("Array contains: [");
	for(int i = 0; i < sizeof(arr)/sizeof(arr[0]); i++)
	{
		printf("%d ",arr[i]);
	}
	printf("]\n\n");

	//calling to max fun
	result = fptr1(arr, sizeof(arr)/sizeof(arr[0]));
	printf("Max element from given array is: %d\n",result);



	dlclose(lib1);

	/////////////////////////////////////////////////////////////////


	lib2  = dlopen("./Min.so", RTLD_LAZY);

	if(lib2 == NULL)
	{
		printf("Unable to open Min.so\n");
		exit(-1);
	}

	fptr2 = dlsym(lib1, "_min");


	if(fptr2 == NULL)
	{
		printf("Unable to get address of min function\n");
		exit(-1);
	}

	//calling to min fun
	result = fptr2(arr, sizeof(arr)/sizeof(arr[0]));
	printf("Min element from given array is: %d\n",result);


	dlclose(lib2);

	exit(EXIT_SUCCESS);
}
```

```c
//definition of max function
//max.c
int _max(int iarr[], int isize)
{
	int imax = iarr[0];

	for(int i = 1; i < isize; i++)
	{
		if(iarr[i] > imax)
		{
			imax = iarr[i];
		}
	}


	return imax;
}
```

```c
//definition of min function
//min.c
int _min(int iarr[], int isize)
{
	int imin = iarr[0];

	for(int i = 1; i < isize; i++)
	{
		if(iarr[i] < imin)
		{
			imin = iarr[i];
		}
	}

	return imin;
}
```

```c
//headers.h


int _min(int iarr[],int isize);
int _max(int iarr[],int isize);
```

----

create .c to .o file using .cmd

```
$gcc -g -c -fPIC max.c min.c
```

this above cmd created 2 files as max.o min.o, we use this two files to create seperate .so file

```
$gcc -g -shared -o min.so min.o
```
creates min.so file

```
$gcc -g -shared -o max.so max.o
```
creates max.so file

now above created .so files we open in our main.c to perform find Max and find Min element from array.