1. When residing in system’s main memory, the program image is divided into (at least) four segments of regions: text (or program), data, heap and stack.
   Explain the difference between the text segment and data segment. Please be precise and concise.

2. Every active UNIX process is in one of the four possible states from the time its being invoked until it terminates. Name the states below, and explain shortly:
   [15 points]
   1. 
   2. 
   3. 
   4. 

5. Draw the tree structure of the processes generated by the program below. The root of the tree should be the process that is created when the program is invoked. Assume that all calls to fork() are successful and ignore the values returned by getpid() and getppid() system calls.
   [20 points]

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

int main()
{
    int a, b, c;

    printf("\nInitial PID: %d  PPID: %d\n", b=getpid(), getppid());
a=fork();
    if(a!=0)
    {
        fork();
        fork();
    }
    else
    {
        a=fork();
        if(a!=0)
            fork();
    }
    if((c=getpid())!=b)
        printf("New PID: %d  PPID: %d\n", c,getppid());
}
```
4. Signal SIGALRM is raised with every call to C library function \texttt{alarm}. The function prototype of \texttt{alarm} is:
\begin{verbatim}
unsigned alarm(unsigned seconds);
\end{verbatim}

When SIGALRM is raised it terminates the process which raises it. What will be the effect of running the following program?

\begin{verbatim}
#include <unistd.h>

main(void) {
    alarm(3);
    for( ; ; ) printf("Still Alive ");
}
\end{verbatim}

\textbf{SOLUTION:}
The program prints \texttt{Still Alive} continually for 3 seconds, and then terminates.

5. Please draw the contents of the stack segment upon the execution of each function call in the program below:

\begin{verbatim}
#include <stdio.h>

main()
    a();
    b();

int a()
    c();

int b()
    return(0);

int c()
    return(0);}
\end{verbatim}