PROGRAM CODE

a.

```
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>
#include <sys/types.h>

void fibonacci(int n) {
	int x = 0, y = 1;

	while(y < n) {
		printf("%d ", y);
		int temp = y;
		y = x + y;
		x = temp;
	}
}

void prime(int n) {
	for (int i = 2; i < n; i++) {
		int flag = 0;

		for (int j = 2; j < i / 2; j++) {
			if (i % j == 0)
				flag = 1;
		}

		if (flag == 0)
			printf("%d ", i);
	}
}

void main() {
	int n;

	printf("Enter the value of N: ");
	scanf("%d", &n);

	pid_t pid = fork();

	if(pid == 0) {
		printf("Child (Fibonacci): ");
		fibonacci(n);
		printf("\n");
	}
	else if (pid == -1)
		printf("Child could not be created!\n");
	else {
		wait(NULL);
		printf("Parent (Prime): ");
		prime(n);
		printf("\n");
	}
}
```

b.

```
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>

void main() {
	int n;

	printf("Enter the value of N: ");
	scanf("%d", &n);

	printf("\nParent pid (Main process) %d at level 0\n", getpid());

	for(int i = 1; i <= n; i++)
		if(fork() == 0) // CHILD 1
			printf("Child pid %d from parent pid %d at level %d\n", getpid(), getppid(), i);
		else if(fork() == 0) // CHILD 2
			printf("Child pid %d from parent pid %d at level %d\n", getpid(), getppid(), i);
		else { // PARENT
			wait(NULL);
			i = n+1;
		}
}
```

c.

```
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>

void main() {
	printf("A: %d\n", getpid());
	if (fork() == 0) {
		printf("B: %d forked by %d\n", getpid(), getppid());
		if (fork() == 0) {
			printf("D: %d forked by %d\n", getpid(), getppid());
			if (fork() == 0) {
				printf("H: %d forked by %d\n", getpid(), getppid());
				if (fork() == 0) {
					printf("I: %d forked by %d\n", getpid(), getppid());
				} else
					wait(NULL);
			} else
				wait(NULL);
		} else if (fork() == 0) {
			printf("E: %d forked by %d\n", getpid(), getppid());
		} else if (fork() == 0) {
			printf("F: %d forked by %d\n", getpid(), getppid());
		} else
			wait(NULL);
	} else if (fork() == 0) {
		printf("C: %d forked by %d\n", getpid(), getppid());
```

```
                if (fork() == 0) {
                        printf("G: %d forked by %d\n", getpid(), getppid());
                } else
                        wait(NULL);
        } else
                wait(NULL);
}
```

SAMPLE OUTPUTS

a.

Enter the value of N: 100

Child (Fibonacci): 1 1 2 3 5 8 13 21 34 55 89

Parent (Prime): 2 3 4 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97

b.

Enter the value of N: 4

Parent pid (Main process) 7250 at level 0

Child pid 7251 from parent pid 7250 at level 1

Child pid 7252 from parent pid 7250 at level 1

Child pid 7253 from parent pid 7251 at level 2

Child pid 7254 from parent pid 7251 at level 2

Child pid 7255 from parent pid 7252 at level 2

Child pid 7256 from parent pid 7253 at level 3

Child pid 7258 from parent pid 7252 at level 2

Child pid 7259 from parent pid 7253 at level 3

Child pid 7260 from parent pid 7255 at level 3

Child pid 7261 from parent pid 7256 at level 4

Child pid 7262 from parent pid 7254 at level 3

Child pid 7257 from parent pid 7254 at level 3

Child pid 7263 from parent pid 7255 at level 3

Child pid 7265 from parent pid 7256 at level 4

Child pid 7264 from parent pid 7258 at level 3

Child pid 7272 from parent pid 7259 at level 4

Child pid 7273 from parent pid 7260 at level 4

Child pid 7274 from parent pid 7262 at level 4

Child pid 7275 from parent pid 7257 at level 4

Child pid 7270 from parent pid 7257 at level 4

Child pid 7271 from parent pid 7263 at level 4

Child pid 7277 from parent pid 7264 at level 4

Child pid 7269 from parent pid 7258 at level 3

Child pid 7276 from parent pid 7263 at level 4

Child pid 7266 from parent pid 1625 at level 4

Child pid 7279 from parent pid 7269 at level 4

Child pid 7267 from parent pid 1625 at level 4

Child pid 7278 from parent pid 7264 at level 4

Child pid 7268 from parent pid 7262 at level 4

Child pid 7280 from parent pid 1625 at level 4

c.

A: 8778

B: 8779 forked by 8778

C: 8780 forked by 8778

D: 8781 forked by 8779

E: 8782 forked by 8779

G: 8783 forked by 8780

F: 8785 forked by 8779

H: 8784 forked by 8781

I: 8786 forked by 8784