ARMAssemblyInt:BigEx:E

main second loop

```
for (toTest = 2; toTest < numsToTest; toTest++)
   if (checkNum(toTest, primeSet))
      totalPrimes++;
```

```
038      mov     r3, #2
039      str     r3, [fp, #-12]
040      b     .L4
041 .L6:
042      ldr     r0, [fp, #-12]
045      mov     r1, sp
046      bl    checkNum
047      mov     r3, r0
048      cmp     r3, #0
049      beq     .L5
050      ldr     r3, [fp, #-8]
051      add     r3, r3, #1
052      str     r3, [fp, #-8]
053 .L5:
054      ldr     r3, [fp, #-12]
055      add     r3, r3, #1
056      str     r3, [fp, #-12]
057 .L4:
058      ldr     r2, [fp, #-12]
059      ldr     r3, [fp, #-20]
060      cmp     r2, r3
061      blt     .L6
```

Main third loop

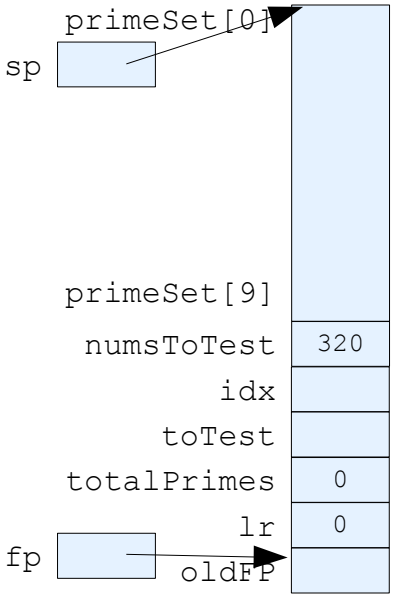
```
printf("Found %d primes:", totalPrimes);
for (toTest = 0; toTest < numsToTest; toTest++)
   if (primeSet[toTest/bitsPerInt] >> (toTest % bitsPerInt) & 0x1)
      printf(" %d", toTest);
```

```
062      ldr     r0, =.LC0
063      ldr     r1, [fp, #-8]
064      bl    printf

065      mov     r3, #0
066      str     r3, [fp, #-12]
067      b     .L7
068
.L9:
069      ldr     r3, [fp, #-12]
070      mov     r3, r3, lsr #5
071      mov     r3, r3, asl #2
072      sub     r2, fp, #4
073      add     r3, r2, r3
074      ldr     r2, [r3, #-56]
075      ldr     r3, [fp, #-12]
076      and     r3, r3, #31
077      mov     r3, r2, lsr r3
078      and     r3, r3, #1
079      cmp     r3, #0
080      beq     .L8

081      ldr     r0, .L10+4
082      ldr     r1, [fp, #-12]
083      bl    printf
084 .L8:

085      ldr     r3, [fp, #-12]
086      add     r3, r3, #1
087      str     r3, [fp, #-12]
088 .L7:
089      ldr     r2, [fp, #-12]
090      ldr     r3, [fp, #-20]
091      cmp     r2, r3
092      blt     .L9
```




```
printf("\n");

return 0;
```

```
093      mov     r0, #10
094      bl      putchar
095      mov     r0, #0
096      sub     sp, fp, #4
097      ldmfd     sp!, {fp, pc}
```