```
#include <stdio.h>           /* For standard input/output */
#include <stdlib.h>          /* For memory allocation and deallocation */
#include  <assert.h>         /* For input validation */
#include <ctype.h>           /* For alphabet manipulation */

#define EOS '\0'       /* For end of strings

/********************************* MAIN *************************************/

void TF(int num);
void ReverseString(*char str);
int FindHeavyCookie();
int FindOneOfThree(*int group);

int main()
{
      //test TF
      TF(6);
      TF(10);
      TF(15);
      return 0;
}

/* a. function to print numbers from 1 to num as follow: */
void TF(int num)
{
      if(num % 3 == 0)
      {
            printf("T\n");
      }
      if(num % 5 == 0)
      {
            printf("F\n");
      }
      if(num % 3 == 0 && num % 5 == 0)
      {
            printf("TF\n");
      }
      else
      {
            printf("%d\n", num);
      }
}

/* b. function to reverse string & lower all capitals alphbets */
void ReverseString(*char str)
{
      assert(NULL !=  src);   /*validate inputs*/
      unsigned int i = 0;     /*counter for 1st run*/
      unsigned int j = 0;     /*counter for 2nd run*/

      char* rvrsd = (char*)malloc((i + 1) * sizeof(char)); /*counter for 1st run*/

      while(! i)
      {
            *(rvrsd+j) = tolower( *(str+i) );
            --i;
            ++j;
      }
```

```
      *rvrsd = *(str+j+1);
      *(rvrsd+j+1) = EOS;

      str=rvrsd; //assign origin
      free(rvrsd);
      *rvrsd=NULL;
}

/* c. function find one heavy cookie out of 9, with only 2 weighing chanses */
int FindHeavyCookie()
{
      int cookies[9]={1,1,1,1,2,1,1,1,1};
      int i=0;
      int f_third[3], s_third[3], t_third[3];
      int f_sum = 0, s_sum = 0, t_sum = 0;
      int group=0; //group can be 1 or 2 or 3;
      int cookie=0;
      for(; i<3, i++)
      {
            f_third[i] = cookies[i+0];
            s_third[i] = cookies[i+3];
            t_third[i] = cookies[i+6];
            ++i;
            f_sum += f_third[i];
            s_sum += s_third[i];
            t_sum += t_third[i];
      }

      //1st weigh:
      if(f_sum == s_third) group=3;
      if(s_sum == t_third) group=1;
      if(t_sum == f_third) group=2;
      //2nd weigh:
      switch (group)
      {
            case 1:
            cookie=FindOneOfThree(f_third);
             break;
            case 2:
            cookie=FindOneOfThree(s_third);
             break;
            case 3:
            cookie=FindOneOfThree(t_third);
             break;
      }
      printf("Heaviest Cookie is # %d\n", group+cookie);
      return group+cookie;
}

/* c. auxilary for FindHeavyCookie() */
int FindOneOfThree(*int group)
{
      if(group[0] == group[1])     return 2;
      if(group[1] == group[2])     return 3;
      return 1;
}
```