```
#include <stdio.h>
#include <string.h>
int run_through(int num, char **a)
{
        Int i;
        Int check=0;
        For(i=0;i<num;i++)
        {
                printf("%s\n", *(a+i));
                if(!strcmp(*(a+i), "filename"))
                {
                        check=1;
                }
        }
        return  check;
}
char**  find_filename(int n, char **b)
{
        Int i;
        Int check=0;
        for(i=0;i<n;i++)
        {
                if(!strcmp(*b,  "filename"))
                {
                        b++;
                        break;
                }
                b++;
        }
        return  b;
}
Int main(int argc, char **argv)
{
        Int choice;
        Char **hold;
        if(!(argc<6))
        {
                printf("Not happening.\n");
        }
        else
        {
                choice=run_through(argc, argv);
                if(choice==0)
                {
                        printf("Here.\n");
                }
                else
                {
                        hold=find_filename(argc, argv);
                        printf("The filename is %s!\n", *hold);
                }
        }
}
```

1. False. argc and argv are passed as an argument in the function find_filename.
2. True. Because although argc increasing by 1, if the argc is less than 6 you could have any valid command line parameters and the function would execute.
3. False. The value of a does not change in the function
4. True. They all have an int and a double pointer.

5. False. If here printed to the screen, then the hold is not changed.
6. True. Because argc is passed in as the most number of times the for loop will run.
7. False. since hold is dereferenced, then it would print out the name of the file and not the address of it.
8. False. The function run_through's return value cannot always be found in argv since they are not correlated.
9. The value of b does change because of the b++ increment.
10. False. Because argc is greater than 6, the value of choice would not be equal to 0.