I loaded the binary into IDA pro and saw the following

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  char s; // [sp+10h] [bp-100h]@1
  char v5; // [sp+11h] [bp-FFh]@7
  char v6; // [sp+12h] [bp-FEh]@2
  char v7; // [sp+13h] [bp-FDh]@4
  char v8; // [sp+14h] [bp-FCh]@5
  char v9; // [sp+15h] [bp-FBh]@8
  char v10; // [sp+16h] [bp-FAh]@3

  printf("Password, please: ", argv, argv);
  __isoc99_scanf("%255s", &s);
  if ( strlen(&s) != 7 || v6 != 52 || v10 != 98 || v7 != 100 || v8 != 100 || s != 109 || v5 != 117 || v9 != 49 )
    puts("Incorrect");
  else
    puts("Correct");
  return 0;
}
```

So, we can already see that it checks that the string is 7 long and checks that the string is 52,98,100 which are known ascii decimals. After some definitions it looks like this

```
char *password; // [sp+10h] [bp-100h]@1

printf("Password, please: ", argv, argv);
__isoc99_scanf("%255s", &password);
if ( strlen((const char *)&password) != 7
  || BYTE2(password) != '4'
  || BYTE6(password) != 'b'
  || BYTE3(password) != 'd'
  || BYTE4(password) != 'd'
  || (_BYTE)password != 'm'
  || BYTE1(password) != 'u'
  || BYTE5(password) != '1' )
{
  puts("Incorrect");
}
else
{
  puts("Correct");
}
return 0;
```

Now its even more clear, password is “mu4dd1b”