HW 1

1.

(a)

```
1   .file  "foo.c"
2       .text
3   .globl fact
4       .type  fact,@function
5   fact:
6       pushl  %ebp             # 将原ebp值入栈
7       movl   %esp, %ebp       # 把esp值赋给ebp，此时ebp作为新栈的栈底指针
8       subl   $4, %esp         # 给局部变量在栈内分配空间（4字节）
9       cmpl   $0, 8(%ebp)
10      jg .L2                  #比较n与0，若 n>0，跳转至L2
11      movl   $1, -4(%ebp)
12      jmp    .L1              # 否则将1存储在局部变量位置，跳转至L1
13  .L2:
14      subl   $12, %esp        # 给局部变量在栈内分配空间（12字节）
15      movl   8(%ebp), %eax    # 将n加载到寄存器 eax 中
16      decl   %eax             # 将eax值-1
17      pushl  %eax             # 然后将其入栈
18      call   fact             #递归调用fact
19      addl   $16, %esp        #清理调用后的栈空间
20      imull  8(%ebp), %eax    #将调用后的返回结果 (n-1)! 与 n 相乘，存储在寄存器 eax 中
    eax中
21      movl   %eax, -4(%ebp)   # 把结果存在栈内局部变量所在位置
22  .L1:
23      movl   -4(%ebp), %eax   # 恢复栈帧
24      leave                   #返回值
25      ret
```

(b)

- 函数参数通常通过栈传递，该程序中参数n位于 8(%ebp)处，通过访问该位置来获得n的值.
- if (n <= 0) return 1; 对应汇编段落是：

```
    jmp .L1
    ...
.L1 movl    -4(%ebp), %eax
    leave
    ret
```

2.

(2.1)

变量p是一个函数，整个声明表示p函数接受一个int型参数x，并返回一个指向包含20个函数指针的数组的指针，这些函数指针可以接受一个int指针类型的参数y并返回一个int类型的值.

(2.2)

```
iii = 1000 @: 0x6ffdfc
ii = 1000 @: 0x6ffdfc
pii points to 0x6ffdfc with value = 1000 @: 0x6ffdf0
pr points to 0x6ffdfc with value = 1000 @: 0x6ffdf0
```