1.

Step1:CPU 初始化，到 0xFFFF0 BIOS，进行 BIOS 的初始化，功能有硬件自检（内存等）、使用系统配置表找到指定外部设备的系统，并将 MBR 从磁盘读到 CPU 中去

Step2:运行 MBR，跳到活动分区的引导扇区上去（通过跳转指令，跳转指令与 CPU 的设计有关）找到启动代码

Step3:从上面分区的启动代码，找到 Boot Loader。Boot Loader 就是在操作系统内核运行之前运行的一段小程序。通过这段小程序，可以初始化硬件设备、建立内存空间的映射图，从而将系统的软硬件环境带到一个合适的状态，以便为最终调用操作系统内核做好一切准备。

Step4:加载操作系统到 CPU，将控制权交给操作系统镜像

2.

```cpp
#include<stdio.h>
#include<stdlib.h>
int a = 0;
int main()
{
    int i = 0;
    int *t = new int;
    //t = (int *)malloc(sizeof(int));
    char *p;
    p = (char *)malloc(20);
    printf("代码段：%p  \n数据段：%p  \n"
           "栈：%p  \n堆：%p\n",&main,&a,&i,p);
    return 0;
}
```

```
C:\Users\20816\Desktop\treasure_chest\cc\try.exe
代码段：0000000000401530
数据段：0000000000434030
栈：000000000065FE0C
堆：000000000001C14D0

--------------------------------
Process exited after 0.9466 seconds with return value 0
请按任意键继续. . .
```

3.

对函数的计算公式：

(sign(A) + S) ≫ 2

若 A = 0，S = 00400700，则结果为 1001c0

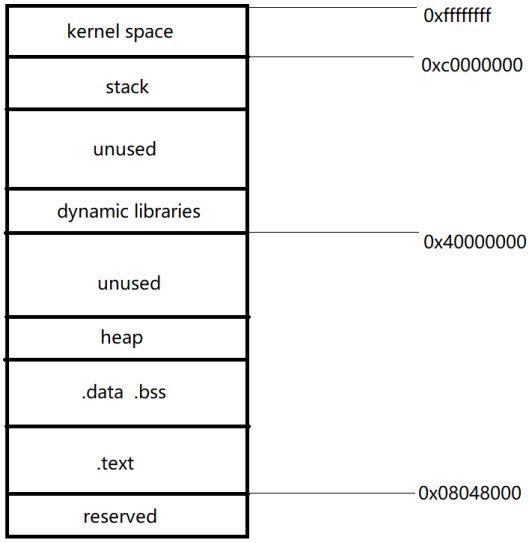
对全局变量的计算公式：

高 16 位的类型为 R_MIPS_HI16，计算公式为((AHL + S)-(short)(AHL + S))>>16，若 AHL 为 0，S 为 00410a1c，结果为 41

低 16 位地址的类型为 R_MIPS_LO16，计算公式为 AHL+S，

若 AHL 为 0， S 为 00410a1c。这里只保留 16 位，因此，结果为 0a1c

4.



代码段.text 存放函数指令，位于内存的 0x08048000
数据段.data.存放已初始化的全局变量和静态变量，bss 段.bss 存放未初始化的全局变量和静态变量，.data 和.bss 位于.text 之后
堆位于.data 和.bss 之后
栈位于 0xc0000000,增长方式向下增长