6.1 CRAY为一个过程名

ENDP CRAY 改为 CRAY ENDP

6.6. SKIPLINES:

```
    MOV  CX,AX      ;行数存入CX
    MOV  AH,2       ;功能02为显示字符
    MOV  DL,0AH     ;DL存入换行符
LOOP1:
    INT  21H        ;DOS调用实现显示字符
    LOOP LOOP1      ;循环行缩进、CX次
    RET             ;返回
```

7.1

```
CLRB MACRO  N,ADDR
    MOV  CX,N
    CLD
    MOV  AL,' '     ;将空格的ASCII存入AL
    LEA  DI,ADDR    ;取首地址
    REP  STOSB      ;全部用空格代替
ENDM
```

7.2

```
WAGES MACRO RATE, HOUR
      MOV AL, HOUR
      MUL RATE          ;计算工资,存入AX
      MOV BX, AX
      MOV AX, HOUR
      DIV 10
      MUL 3
      ADD BX, AX
      MOV WAG, BX
    ENDM
```

展开宏调用:

```
MOV AL, 42
MUL R1
MOV BX, AX
MOV AX, 42
DIV 10
MUL 3
ADD BX, AX
MOV WAG, BX
```

题目 1：

1. IP 相对寻址允许程序使用当前指令地址作为基址，并加上一个偏移量来访问内存中的数据。通过使用 IP 相对寻址，可以减少代码大小，因为不需要显式地指定绝对地址。这种寻址方式使得编译后的代码可以在不同的内存位置运行而无需修改，提高了软件的灵活性和可移植性。

2. SP 相对寻址允许程序使用当前栈顶地址作为基址，并加上一个偏移量来访问栈上的数据。在函数调用过程中，通常会将返回地址压入栈中。SP 相对寻址便于快速访问这个返回地址和其他栈帧信息，从而实现高效的过程调用和返回机制。SP 相对寻址有助于快速定位并恢复栈状态，以进行错误处理和资源清理。

题目 2：

（1）理想的编译器可以在编译时直接计算出结果，并将所有对 fib(5) 的调用替换为这个已知的结果值，避免额外的计算。

编译器也可以通过记忆化的方法来保存已经计算过的 fib(5)，在整个程序中最多只计算一次。

（2）实际的编译器受限于静态分析的能力，可能无法准确地预测所有的运行时行为。特别是当函数调用形式复杂或者存在间接调用时，编译器很难确定哪些调用是恒定的，哪些变量的值是可以提前确定的。

过度的优化可能会显著增加编译时间，过度特定化的优化可能会导致在其他输入下的性能下降。

题目 3：

1. MASM 宏处理器宏定义使用 MACRO 关键字开始，ENDM 关键字结束，宏展开时，宏处理器会将宏体中的指令插入到源代码中，并替换参数。MASM 宏处理器提供了更强大的控制流指令，如条件宏、循环宏等。可以生成复杂的汇编代码块，适合编写复杂的汇编程序。
2. C 语言宏处理器宏定义使用#define 预处理指令，实现简单的文本替换，也可以带参数。不支持复杂的控制流指令，但可以通过条件编译语句来实现一些控制逻辑。

滥用宏处理器会导致代码可读性和可维护性降低，宏定义中的错误难以调试。

题目 4：

使用递减栈时，堆从低地址向高地址增长，栈从高地址向低地址增长。这种布局可以有效地利用内存空间，减少堆和栈之间的冲突。

题目 5：

链接时优化允许编译器在链接阶段对整个程序进行优化，而不仅仅是在编译单个源文件时进行优化。

链接时优化可以跨越多个源文件进行函数内联，减少函数调用开销，提高性能，可以识别并删除在整个程序中从未使用的函数和变量，减少二进制文件的大小，还可以在整个程序范围内进行常量传播和折

叠，进一步优化代码。链接时优化共享不同模块之间的类型信息，帮助编译器生成更高效的代码。