理论部分答案

Problem1

%rbx %rbp %r12 %r13 %r14 %r15

Problem2

endbr64

movq %rsp (%rdi)

movq $0x0 %rax

retq

Problem3

被调用的函数没有使用到寄存器%rbp。

Problem4

将%rax设置为函数B应有的返回值，将被调用者保存寄存器从栈中取出并还原，将栈指针%rsp设置为函数 A 调用 B 之前的值。

Problem5

C++的 try-catch 语法规则

C++标准库 `stdexcept` 定义了异常类型。

使用如下语句即可捕捉在 `try` 之后的语句块中的异常，在 `try` 中的语句执行遇到异常后就会立刻跳转执行 `catch` 中的语句，示例如下：

```
try{
    ……
}
catch(exception& e){
    cout << e.what();
    ……
}
```

Problem6

在 `try` 执行的时候保存该处的上下文，之后执行 `try` 之后的语句块中如果遇到了异常就恢复之前保存的上下文，利用保存上下文函数不同的返回值控制函数开始执行 `catch` 处的代码块

Problem7

输出为：

```
0
1
2
3
4
5
6
7
8
9
```

Problem8

在调用 generator 函数时保存该处的上下文，并恢复到执行 generator 函数时保存的上下文，在执行 generator 函数时如果遇到 yield 函数也保存该处的上下文并恢复到 generator 调用者

的上下文。另外，初次调用 generator 函数时需要手动配置一份上下文，因为之前没有 yield 保存那时的上下文。
Problem9
将在 generator 中遇到的异常保存到某个其调用者可以访问的变量，之后将上下文恢复到 generator 的调用者之后再把该异常抛出。

lab 的实现思路
用__ctx_record 记录所有被调用者保存寄存器的值，栈指针的值以及函数返回时的跳转地址，在__ctx_recover 中恢复所有被调用者保存寄存器的值和栈指针的值，再令其跳转到记录的返回点。如此可以通过 test1 和 test2。
把 try　catch 用 if else 实现，将__ctx_record 的返回值赋值给 error，如果 error 为 0 则执行 catch 之后的语句块，否则该处是由__ctx_recover 恢复上下文得到的，应直接进入 catch 部分，保存的上下文通过__eh_push 压入异常处理栈，__eh_pop 将栈顶的上下文弹出。由于我在__ctx_type 结构的首地址处就保存了一个指针指向下一个__ctx_type 类型的变量，所以我的两个指针之间并没有偏移量。如果该处保存的上下文由于 throw 被恢复了，则已经从异常处理栈中弹出了，不需要进行额外处理，如果并没有被恢复，则在 error 结束其生存期之后说明 catch 之后的代码块已经正常结束，此时在 clean-up function 中将其从异常处理栈中弹出即可。如此可以通过 test3 和 test4。
yield 和 send 的保存上下文和恢复上下文的代码依照 kieraylab 文件中给出的做法写代码即可，generator 函数则有些棘手，手动配置上下文时将需要跳转到的函数语句的地址赋给__ctx_type 的 jmp 成员，除此之外还需要把参数 arg 传给%rdi，于是再返回修改__ctx_recover 和__ctx_record，为 struct __ctx_type 中新增一成员%rdi，并且在函数 generator 初始化时为其赋值为 arg，这样就手动配置好了上下文。由于栈指针在保存数据是是不断减小的于是把栈指针设定为新分配空间的顶部（保持是 16 的倍数）。之后在该栈指针上方放置新定义函数__throw_error 的地址，该函数会抛出错误__ERR_GENEND，这样可以使得 generator 返回之后可以抛出结束的错误使得该异常能够被处理。在第一次做的时候没有注意到可以在 gen.c 文件中新增加函数，一直不知道怎么让 generator 返回值后抛出错误，在逸夫楼询问助教之后才注意到文件中的这句话，是自己没有认真看题目要求的原因导致卡壳。之后顺利通过 test5,和 test6。

对 lab 中有关内容的思考
感觉助教的 kieraylab 文件写的十分清晰，只要耐心仔细阅读不仅可以学到很多新东西也可以一步一步的解开 lab 的各个 test，很有意思。由于助教的引导工作十分出色于是在思路方面也基本没有障碍。解完整个 lab 之后感觉对于程序在机器级层面的运行方式又多了很多认识（不知道怎么描述），大概就是那种课堂上学的课本知识成功得到了验证的那种感觉，对书本知识掌握的也更加透彻了。

对本门课程的建议
话说这个课程虽然只是 3 学分但是花费的时间真的很多了，相比于其他老师的 ics 课程也是明显感觉要花更多的时间(我的两个室友都能大概以我周花费时间的一半到三分之二就能完成他们 ics 的每周的任务，而且我的能力等各方面甚至还要比他们强一些些子），不过收获也很大就是说。悄悄问，能不能申请一下 40%的 A（探头）。我的其他任务比其他同学更重的课程（如孙未未数据结构还有集合与图论的荣誉课）都是超出 30%A 限制的（偷笑）。