Homework 3 – CS 0449

Question 1: Assuming AT&T Syntax, which of the following x86 instructions is both correctly written and will compute %rax = %rdi * 5 ?

A: mov (%rdi), %rax

B: mov %rdi * 5, %rax

C: lea (, %rdi, 5), %rax

D: lea (%rdi, %rdi, 4), %rax

Answer: **D**

Question 2: Which of the following would you highlight as an example of a CISC processor instruction?

A: Adding two numbers

B: Finding length of a C string

C: Jumping to an address

D: 64-bit XOR Operation

Answer: **D**

Question 3: Of the following, what x86 register might you expect a compiler to use when utilizing a variable declared as a short when SHORT_UMAX is defined as $2^{16} - 1$?

A: %rax

B: %eax

C: %ax

D: %al

Answer: **C**

Question 4: Consider the x86-64 Linux C ABI mentioned in our slides. How many items (not byte size) are written to the stack as part of this (otherwise nonsense) function's activation frame to account for arguments and variables? (Ignore the return address and any potential preserved registers, such as %rbp.)

```
int main(char x, char y, int i, int j, const char* s, char* d, size_t length) {
   char* tmp = s;

   while(*d++ != '\0') {
      *d = x;
   }

   int num = 0;

   while(*s++ != '\0' && num < length) {
      *d++ = *(s - 1);
      num++;
   }

   tmp[0] = y;

   return num;
}
```

Answer: **X,y,I,j,s,d,length,tmp,num**

Question 5: Consider the following x86 assembly. What is the value is %rax at the end of its execution? (An int, here, is 32-bits, you can specify your answer in base 10.)

```
.data
arr: .int 1, -2, 6, -4, 11

.text
_start:
        lea  (arr), %rbx  //rbx = addr to arr
        mov  $4, %rdi // rdi = 4
        lea  (%rbx, %rdi, 4), %rax // rax = &arr[4]
        movl (%rax), %eax // eax = &rax
```

Answer: 11

Submission:

Please modify this document and answer in the provided spaces and submit your completed document as a PDF to Gradescope. You may write in your answers and scan them in. Or carefully modify this document in Word and export to PDF.