CS – Übung 10

150 Punkte, davon 50 Extrapunkte.

U 10.1)

Gegeben ist folgender C-Code:

```c
#include <stdio.h>
#include <stdlib.h>

// Function prototypes
long calculateSumAndMultiply(long, long, long, long,
                             long, long, long, long, long *);
long multiply(long, long, long *);

int main() {
    long rdi = 1, rsi = 2;
    long sum = 0; // Define sum in main
    long result = calculateSumAndMultiply(rdi, rsi, 3, 4, 5, 6, 7, 8, &sum);
    printf("Result: %ld\n", result + rdi + rsi);
    return 0;
}

long calculateSumAndMultiply(long l1, long l2, long l3, long l4,
                             long l5, long l6, long l7, long l8, long *pSum){
    *pSum = l1 + l2 + l3 + l4 + l5 + l6 + l7 + l8;
    return multiply(2, 3, pSum);
}

long multiply(long mult1, long mult2, long *pSum){
    return *pSum * mult1 * mult2;
}
```

Übersetzen Sie den C-Code in GNU-Assembler und beachten Sie alle Calling Conventions. Sie können davon ausgehen, dass Sie nur Register sichern müssen, die im Caler tatsächlich später noch gebraucht oder im Callee tatsächlich überschrieben werden – also nicht (vorsichtshalber) alle.

Der Fokus in dieser Aufgabe liegt hauptsächlich auf dem korrekten Umgang mit dem Stack.