Übung 02

Table of Contents

1. Lösungsidee

1.1. Teil A

1.1.1. Simulator

Der Simulator ist relativ einfach gehalten, er erhält bei der Initialisierung Events, welche direkt in die `priority_queue` eingefügt werden. Mithilfe von `run` wird die ganze Queue ausgeführt, bis diese leer ist oder ein Event terminiert. Die `step` Methode führt immer nur ein Event aus und gibt zurück ob die Queue leer ist oder ein Event terminiert hat. Als Zeiteinheit wird eine Timestamp genutzt.

1.1.2. Event

Das Basis Event hat drei pure virtual Functions:

- `bool terminates` gibt an ob ein Event terminiert.
- `vector<shared_ptr> execute()` führt daas Event aus und gibt folge Events zurück
- `string details()` gibt Eventdetails zurück, häufig einfach der Name und Timestamp.

1.2. Teil B

1.2.1. Machine Event

Das Basis Event für die Maschinen. Hat einen Pointer auf den momentanen Status, also ob Maschinen angehalten wurden, alle Produkte welche gerade im Puffer liegen und alle Produkte welche gerade im Lager liegen.

Product

Ein Product wird durch dessen Erstellungszeitpunkt und Fertigstellungszeitpunkt ausgezeichnet.

1.2.2. Start Event

Initialisiert alle wichtigen Eigentschaften, wie State und Abbruchkriterien.

1.2.3. Management Event

Wird Regelmässig ausgeführt. Überprüft ob Maschinen angehalten wurden und ob Abbruchkriterien erfüllt sind. Sollte die der Fall sein, wird das End Event ausgelöst.

1.2.4. End Event

Wirkt terminierend. Gibt auch noch die Details zur Produktion aus.

1.2.5. Machine A Event

Erstellt ein Produkt und fügt dieses im Puffer ein.

1.2.6. Machine B Event

Verschiebt das Produkt vom Puffer ins Lager.

2. Source Code

Listing 1. simulator.h

```
//
// Created by florian on 27.03.21.
//

#pragma once

#include <iostream>
#include <queue>
#include <utility>
#include <unistd.h>
#include "events/event.h"

namespace des {
  class simulator {
  public:

    using event_ptr = std::shared_ptr<event>;

    /**
     *
     * @param init_list
     * @param wait_time
     */
    simulator(std::initializer_list<event_ptr> init_list) {
      for (const auto &item : init_list) {
        this->events_.push(item);
      }
    }

    /**
     * Execute all events until queue is empty
     */
    void run() {
      while (execute_top_event()) {
        sleep(simulator::wait_time_);
      }
      this->stop();
    }

    /**
     * Execute exactly one event
     * @returns true if the simulations was terminated
     */
    bool step() {
      const bool stopped = !execute_top_event();
      if (stopped) {
        this->stop();
      }
      return stopped;
    }

  private:
    std::priority_queue<event_ptr,
        std::vector<event_ptr>,
        event::greater_than_comparator> events_;

    static const unsigned int wait_time_ = 1;

    /**
     * Executes the top event in the queue
     * @returns false if there was no event or if an event terminated the simulation
     */
    bool execute_top_event() {
      if (this->events_.empty()) {
        return false;
      }
      event_ptr e = this->events_.top();
      this->events_.pop();

      std::cout << (*e) << std::endl;
      std::vector<event_ptr> new_events = e->execute();
      for (const auto &item: new_events) {
        this->events_.push(item);
      }
      return !e->terminates();
    }
    void stop() const {
      std::cout << "Terminating Simulation!" << std::endl;
    }
  };
}
```

Listing 2. event.h

```
//
// Created by florian on 27.03.21.
//

#pragma once

#include <ostream>
#include <chrono>
#include <memory>
#include <vector>

namespace des {
  class event {
  public:

    using event_ptr = std::shared_ptr<event>;

    friend std::ostream &operator<<(std::ostream &os, const event &e) {
      return os << e.details();
    }

    friend bool operator<(const event &e1, const event &e2) {
      return e1.execution_time_ < e2.execution_time_;
    }

    friend bool operator>(const event &e1, const event &e2) {
      return e1.execution_time_ > e2.execution_time_;
    }

    struct greater_than_comparator {
      bool operator()(const event_ptr & left, const event_ptr & right) {
        return *left > *right;
      }
    };

    explicit event(std::time_t execution_time) :
        execution_time_{execution_time} {}

    virtual std::vector<event_ptr> execute() = 0;

    [[nodiscard]] virtual bool terminates() const = 0;
  protected:
    std::time_t execution_time_;

    [[nodiscard]] virtual std::string details() const = 0;

    /**
     * Simple details function
     * @param name the name of the event
     * @returns the eventname plus the timestamp
     */
    [[nodiscard]] std::string details(const std::string &name) const {
      return name + "@" + std::to_string(this->execution_time_);
    }
  };
}
```

Listing 3. machine_event.h

```
//
// Created by florian on 05.04.21.
//

#pragma once

#include <memory>
#include <utility>

namespace des {
  class machine_event : public event {
  public:

  protected:
    struct product {
      time_t creation_time_ = time(nullptr);
      time_t end_time_ = time(nullptr);
    };
    /**
     * Struct to document the state of the machines
     */
    struct machine_state {
      machine_state() = default;

      bool machine_a_ = false;
      bool machine_b_ = false;

      std::queue<product> buffer_ = std::queue<product>();
      std::vector<product> store_ = std::vector<product>();
    };

    using state_ptr = std::shared_ptr<machine_state>;

    machine_event(time_t t, state_ptr state) :
        event(t),
        state_(std::move(state)) {}

    state_ptr state_;
    static const int max_buffer_ = 10;

    [[nodiscard]] time_t get_random_future_time() const {
      return this->execution_time_ + (rand() % 100);
    }

  };
}
```

Listing 4. start_event.h

```
//
// Created by florian on 06.04.21.
//

#pragma once

#include "machine_event.h"
#include "management_event.h"

namespace des {
  class start_event : public machine_event {
  public:
    start_event(unsigned int max_count_of_products, time_t max_time) :
        machine_event(time(nullptr),
                      std::make_shared<machine_state>()),
        max_count_of_products_{max_count_of_products},
        max_time_{max_time} {}

    [[nodiscard]] bool terminates() const override {
      return false;
    }

    [[nodiscard]] std::vector<event_ptr> execute() override {
      auto vec = std::vector<event_ptr>();
      vec.push_back(std::make_shared<management_event>(time(nullptr), this->state_,
max_count_of_products_, max_time_));
      return vec;
    }

  protected:
    [[nodiscard]] std::string details() const override {
      return event::details("Start Event");
    }

  private:
    unsigned int max_count_of_products_;
    time_t max_time_;
  };
}
```

Listing 5. management_event.h

```
#include <utility>
#include "end_event.h"

//
// Created by florian on 06.04.21.
//
```

```
#pragma once

#include "machine_event.h"
#include "machine_a_event.h"
#include "machine_b_event.h"

namespace des {
  class management_event : public machine_event {
  public:
    explicit management_event(time_t t, state_ptr state, unsigned int
max_count_of_products, time_t max_time) :
        machine_event(t, std::move(state)),
        max_products_{max_count_of_products},
        max_time_{max_time} {}

    [[nodiscard]] std::vector<event_ptr> execute() override {
      auto vec = std::vector<event_ptr>();

      if (this->state_->store_.size() > max_products_ || time(nullptr) >= max_time_)
{
        vec.push_back(std::make_shared<end_event>(this->state_));
      } else {
        if (this->state_->buffer_.size() < machine_event::max_buffer_ && !this-
>state_->machine_a_) {
          this->state_->machine_a_ = true;
          vec.push_back(std::make_shared<machine_a_event>(machine_event
::get_random_future_time(), this->state_));
        }

        if (!this->state_->buffer_.empty() && !this->state_->machine_b_) {
          this->state_->machine_b_ = true;
          vec.push_back(std::make_shared<machine_b_event>(machine_event
::get_random_future_time(), this->state_));
        }

        time_t next_management_event = this->execution_time_ + offset_;
        vec.push_back(std::make_shared<management_event>(next_management_event, this
->state_, this->max_products_, this->max_time_));
      }
      return vec;
    }

    [[nodiscard]] bool terminates() const override {
      return false;
    }

  protected:
    [[nodiscard]] std::string details() const override {
      return event::details("Management Event");
    }

  private:
    static const time_t offset_ = 20;
    unsigned int max_products_;
    time_t max_time_;
  };
}
```

Listing 6. machine_a_event.h

```
//
// Created by florian on 05.04.21.
//

#pragma once

#include <utility>
#include "machine_event.h"


namespace des {
  class machine_a_event : public machine_event {
  public:

    machine_a_event(time_t t, state_ptr state) :
        machine_event(t, std::move(state)) {}


    std::vector<event_ptr> execute() override {
      auto vec = std::vector<event_ptr>();
      if (this->state_->buffer_.size() < machine_event::max_buffer_) {
        product p = product{};
        p.creation_time_ = this->execution_time_;
        this->state_->buffer_.push(p);
      }

      // If buffer is full stop producing
      if (this->state_->buffer_.size() >= machine_event::max_buffer_) {
        this->state_->machine_a_ = false;
      } else {
        vec.push_back(
            std::make_shared<machine_a_event>(machine_event::
get_random_future_time(),
                                              this->state_)
        );
      }
      return vec;
    }

    [[nodiscard]] bool terminates() const override {
      return false;
    }

  protected:
    [[nodiscard]] std::string details() const override {
      return event::details("Machine A Event");
    }
  };
}
```

Listing 7. machine_b_event.h

```
//
// Created by florian on 05.04.21.
//

#pragma once

#include <utility>

#include "machine_event.h"

namespace des {
  class machine_b_event : public machine_event {
  public:

    machine_b_event(time_t t, state_ptr state) :
        machine_event(t, std::move(state)) {}

    std::vector<event_ptr> execute() override {
      auto vec = std::vector<event_ptr>();
      if (!this->state_->buffer_.empty()) {
        auto product = this->state_->buffer_.front();
        product.end_time_ = this->execution_time_;
        this->state_->store_.push_back(product);
        this->state_->buffer_.pop();
      }

      // If buffer is empty stop moving products
      if (this->state_->buffer_.empty()) {
        this->state_->machine_b_ = false;
      } else {
        vec.push_back(
            std::make_shared<machine_b_event>(machine_event::
get_random_future_time(),
                                              this->state_)
        );
      }
      return vec;
    }

    [[nodiscard]] bool terminates() const override {
      return false;
    }

  protected:
    [[nodiscard]] std::string details() const override {
      return event::details("Machine B Event");
    }
  };
}
```

Listing 8. end_event.h

```
//
// Created by florian on 06.04.21.
//

#pragma once

#include <sstream>

namespace des {
  class end_event : public machine_event {
  public:
    explicit end_event(state_ptr state) :
        machine_event(time(nullptr), std::move(state)) {}

    [[nodiscard]] bool terminates() const override {
      return true;
    }

    [[nodiscard]] std::vector<event_ptr> execute() override {
      return std::vector<event_ptr>();
    }

  protected:
    [[nodiscard]] std::string details() const override {
      time_t total = 0;
      for (const auto &product : this->state_->store_) {
        total += product.end_time_ - product.creation_time_;
      }
      unsigned long avg = static_cast<unsigned long>(total) / this->state_->store_
.size();
      std::stringstream stream;
      const std::string separator = "----------------------------";
      stream << event::details("End Event") << std::endl
             << separator << std::endl
             << "Final Stats:" << std::endl
             << "Average Time of Production: " << std::to_string(avg) << "s" << std
::endl
          << separator << std::endl;
      return stream.str();
    }
  };
}
```

Listing 9. main.cpp

```cpp
//
// Created by florian on 27.03.21.
//

#include <cstdlib>
#include "simulator.h"
#include "scenario/start_event.h"

void test_produce_enough_products() {
  des::simulator{
      std::make_shared<des::start_event>(50, time(nullptr) * 2)
  }.run();
}

void test_produce_enough_products_with_step() {
  des::simulator sim{
      std::make_shared<des::start_event>(50, time(nullptr) * 2)
  };
  while(!sim.step()) {
    sleep(1);
  }
}

void test_time_terminates() {
  const time_t in_a_min = time(nullptr) + 60;
  des::simulator{
      std::make_shared<des::start_event>(1000000, in_a_min)
  }.run();
}

void test_time_terminates_with_step() {
  const time_t in_a_min = time(nullptr) + 60;
  des::simulator sim{
      std::make_shared<des::start_event>(1000000, in_a_min)
  };
  while(!sim.step()) {
    sleep(1);
  }
}

int main() {
  srand(time(nullptr));
  test_time_terminates_with_step();
  return EXIT_SUCCESS;
}
```

3. Test-Cases

Anmerkung, die Simulation wartet immer ca 1s um die zeitbasierten Test Cases besser darzustellen

3.1. Ausführen des kompletten Szenarios

Es wird das ganze Szenario mithilfe von `run` ausgeführt. Als Abbruchbendinung wird die Menge der Produkte genutzt. Erwartet wird ein erfolgreicher Durchlauf.

Listing 10. Output

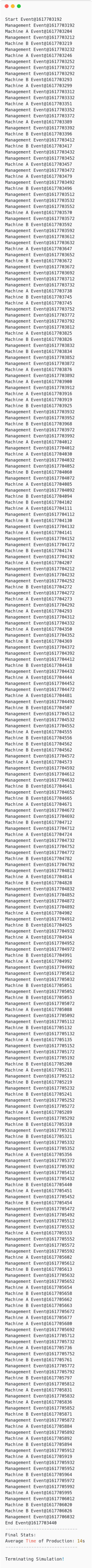
```
Start Event@1617703192
Management Event@1617703192
Machine A Event@1617703204
Management Event@1617703212
Machine B Event@1617703219
Management Event@1617703232
Machine A Event@1617703246
Management Event@1617703252
Management Event@1617703272
Management Event@1617703292
Machine B Event@1617703293
Machine A Event@1617703299
Management Event@1617703312
Management Event@1617703332
Machine A Event@1617703351
Management Event@1617703352
Management Event@1617703372
Machine A Event@1617703389
Management Event@1617703392
Machine B Event@1617703396
Management Event@1617703412
Machine B Event@1617703417
Management Event@1617703432
Management Event@1617703452
Machine A Event@1617703457
Management Event@1617703472
Machine A Event@1617703479
Management Event@1617703492
Machine B Event@1617703496
Management Event@1617703512
Management Event@1617703532
Management Event@1617703552
Machine A Event@1617703570
Management Event@1617703572
Machine B Event@1617703592
Management Event@1617703592
Management Event@1617703612
Management Event@1617703632
Machine A Event@1617703647
Management Event@1617703652
Machine B Event@1617703672
Management Event@1617703672
Management Event@1617703692
Management Event@1617703712
Management Event@1617703732
Machine A Event@1617703738
Machine B Event@1617703745
Machine A Event@1617703745
Management Event@1617703752
Management Event@1617703772
Management Event@1617703792
Management Event@1617703812
Machine A Event@1617703825
Machine B Event@1617703826
Management Event@1617703832
Machine B Event@1617703834
Management Event@1617703852
Management Event@1617703872
Machine A Event@1617703876
Management Event@1617703892
Machine A Event@1617703900
Management Event@1617703912
Machine A Event@1617703916
Machine A Event@1617703919
Machine B Event@1617703925
Management Event@1617703932
Management Event@1617703952
Machine B Event@1617703968
Management Event@1617703972
Management Event@1617703992
Machine A Event@1617704012
Management Event@1617704012
Machine A Event@1617704030
Management Event@1617704032
Management Event@1617704052
Machine B Event@1617704060
Management Event@1617704072
Machine A Event@1617704085
Management Event@1617704092
Machine B Event@1617704094
Machine B Event@1617704102
Machine A Event@1617704111
Management Event@1617704112
Machine B Event@1617704130
Management Event@1617704132
Machine A Event@1617704141
Management Event@1617704152
Management Event@1617704172
```

```
Machine B Event@1617704174
Management Event@1617704192
Machine A Event@1617704207
Management Event@1617704212
Management Event@1617704232
Management Event@1617704252
Machine B Event@1617704272
Management Event@1617704272
Machine A Event@1617704273
Management Event@1617704292
Machine A Event@1617704293
Management Event@1617704312
Management Event@1617704332
Machine A Event@1617704350
Management Event@1617704352
Machine B Event@1617704369
Management Event@1617704372
Management Event@1617704392
Management Event@1617704412
Machine B Event@1617704418
Management Event@1617704432
Machine A Event@1617704444
Management Event@1617704452
Management Event@1617704472
Machine A Event@1617704481
Management Event@1617704492
Machine B Event@1617704507
Management Event@1617704512
Management Event@1617704532
Management Event@1617704552
Machine A Event@1617704555
Machine B Event@1617704556
Machine B Event@1617704562
Machine B Event@1617704562
Management Event@1617704572
Machine A Event@1617704573
Management Event@1617704592
Management Event@1617704612
Management Event@1617704632
Machine B Event@1617704641
Management Event@1617704652
Machine B Event@1617704665
Machine A Event@1617704671
Management Event@1617704672
Management Event@1617704692
Machine B Event@1617704712
Management Event@1617704712
Machine A Event@1617704724
Management Event@1617704732
Management Event@1617704752
Management Event@1617704772
Machine B Event@1617704782
```

```
Management Event@1617704792
Management Event@1617704812
Machine A Event@1617704814
Machine B Event@1617704828
Management Event@1617704832
Management Event@1617704852
Management Event@1617704872
Management Event@1617704892
Machine A Event@1617704902
Management Event@1617704912
Machine B Event@1617704925
Management Event@1617704932
Machine A Event@1617704934
Management Event@1617704952
Management Event@1617704972
Machine B Event@1617704991
Machine A Event@1617704992
Management Event@1617704992
Management Event@1617705012
Management Event@1617705032
Machine B Event@1617705051
Management Event@1617705052
Machine B Event@1617705053
Management Event@1617705072
Machine A Event@1617705088
Management Event@1617705092
Management Event@1617705112
Machine B Event@1617705132
Management Event@1617705132
Machine A Event@1617705135
Management Event@1617705152
Management Event@1617705172
Management Event@1617705192
Machine B Event@1617705200
Machine A Event@1617705211
Management Event@1617705212
Machine B Event@1617705219
Management Event@1617705232
Machine B Event@1617705241
Management Event@1617705252
Management Event@1617705272
Machine A Event@1617705289
Management Event@1617705292
Machine B Event@1617705310
Management Event@1617705312
Machine B Event@1617705321
Management Event@1617705332
Management Event@1617705352
Machine A Event@1617705356
Management Event@1617705372
Management Event@1617705392
Management Event@1617705412
Management Event@1617705432
Machine B Event@1617705440
Machine A Event@1617705451
Management Event@1617705452
Machine B Event@1617705454
Management Event@1617705472
Management Event@1617705492
Management Event@1617705512
Management Event@1617705532
Machine A Event@1617705533
Management Event@1617705552
Management Event@1617705572
Management Event@1617705592
Machine A Event@1617705602
Management Event@1617705612
Machine B Event@1617705613
Management Event@1617705632
Management Event@1617705652
Machine A Event@1617705654
Machine B Event@1617705658
Machine A Event@1617705662
Machine B Event@1617705663
Management Event@1617705672
Machine A Event@1617705677
Machine A Event@1617705680
Management Event@1617705692
Management Event@1617705712
Management Event@1617705732
Machine A Event@1617705736
Management Event@1617705752
Machine B Event@1617705761
Management Event@1617705772
Management Event@1617705792
Machine B Event@1617705797
Management Event@1617705812
Machine A Event@1617705831
Management Event@1617705832
Machine A Event@1617705836
Management Event@1617705852
Machine B Event@1617705871
Management Event@1617705872
Machine A Event@1617705884
Management Event@1617705892
Machine A Event@1617705892
Machine B Event@1617705894
Management Event@1617705912
Machine A Event@1617705919
Management Event@1617705932
Management Event@1617705952
Machine B Event@1617705964
Management Event@1617705972
Management Event@1617705992
```

```
Machine A Event@1617705995
Management Event@1617706012
Machine B Event@1617706020
Machine B Event@1617706026
Management Event@1617706032
End Event@1617703440
----------------------------
Final Stats:
Average Time of Production: 14s
----------------------------

Terminating Simulation!
```



```
Start Event@1617703192
Management Event@1617703192
Machine A Event@1617703204
Management Event@1617703212
Machine B Event@1617703219
Management Event@1617703232
Machine A Event@1617703246
Management Event@1617703252
Management Event@1617703272
Management Event@1617703292
Machine B Event@1617703293
Machine A Event@1617703299
Management Event@1617703312
Management Event@1617703332
Machine A Event@1617703351
Management Event@1617703352
Management Event@1617703372
Machine A Event@1617703389
Management Event@1617703392
Machine B Event@1617703396
Management Event@1617703412
Machine B Event@1617703417
Management Event@1617703432
Management Event@1617703452
Machine A Event@1617703457
Management Event@1617703472
Machine A Event@1617703479
Management Event@1617703492
Machine B Event@1617703496
Management Event@1617703512
Management Event@1617703532
Management Event@1617703552
Machine A Event@1617703570
Management Event@1617703572
Machine B Event@1617703592
Management Event@1617703592
Management Event@1617703612
Management Event@1617703632
Machine A Event@1617703647
Management Event@1617703652
Machine B Event@1617703672
Management Event@1617703672
Management Event@1617703692
Management Event@1617703712
Management Event@1617703732
Machine A Event@1617703738
Machine B Event@1617703745
Machine A Event@1617703745
Management Event@1617703752
Management Event@1617703772
Management Event@1617703792
Management Event@1617703812
Machine A Event@1617703825
Machine B Event@1617703826
Management Event@1617703832
Machine B Event@1617703834
Management Event@1617703852
Management Event@1617703872
Machine A Event@1617703876
Management Event@1617703892
Machine A Event@1617703900
Management Event@1617703912
Machine A Event@1617703916
Machine A Event@1617703919
Machine B Event@1617703925
Management Event@1617703932
Management Event@1617703952
Machine B Event@1617703968
Management Event@1617703972
Management Event@1617703992
Machine A Event@1617704012
Management Event@1617704012
Machine A Event@1617704030
Management Event@1617704032
Management Event@1617704052
Machine B Event@1617704060
Management Event@1617704072
Machine A Event@1617704085
Management Event@1617704092
Machine B Event@1617704094
Machine B Event@1617704102
Machine A Event@1617704111
Management Event@1617704112
Machine B Event@1617704130
Management Event@1617704132
Machine A Event@1617704141
Management Event@1617704152
Management Event@1617704172
Machine B Event@1617704174
Management Event@1617704192
Machine A Event@1617704207
Management Event@1617704212
Management Event@1617704232
Management Event@1617704252
Machine B Event@1617704272
Management Event@1617704272
Machine A Event@1617704273
Management Event@1617704292
Machine A Event@1617704293
Management Event@1617704312
Management Event@1617704332
Machine A Event@1617704350
Management Event@1617704352
Machine B Event@1617704369
Management Event@1617704372
Management Event@1617704392
Management Event@1617704412
Machine B Event@1617704418
Management Event@1617704432
Machine A Event@1617704444
Management Event@1617704452
Management Event@1617704472
Machine A Event@1617704481
Management Event@1617704492
Machine B Event@1617704507
Management Event@1617704512
Management Event@1617704532
Management Event@1617704552
Machine A Event@1617704555
Machine B Event@1617704556
Machine B Event@1617704562
Machine B Event@1617704562
Management Event@1617704572
Machine A Event@1617704573
Management Event@1617704592
Management Event@1617704612
Management Event@1617704632
Machine B Event@1617704641
Management Event@1617704652
Machine B Event@1617704665
Machine A Event@1617704671
Management Event@1617704672
Management Event@1617704692
Machine B Event@1617704712
Management Event@1617704712
Machine A Event@1617704724
Management Event@1617704732
Management Event@1617704752
Management Event@1617704772
Machine B Event@1617704782
Management Event@1617704792
Management Event@1617704812
Machine A Event@1617704814
Machine B Event@1617704828
Management Event@1617704832
Management Event@1617704852
Management Event@1617704872
Management Event@1617704892
Machine A Event@1617704902
Management Event@1617704912
Machine B Event@1617704925
Management Event@1617704932
Machine A Event@1617704934
Management Event@1617704952
Management Event@1617704972
Machine B Event@1617704991
Machine A Event@1617704992
Management Event@1617704992
Management Event@1617705012
Management Event@1617705032
Machine B Event@1617705051
Management Event@1617705052
Machine B Event@1617705053
Management Event@1617705072
Machine A Event@1617705088
Management Event@1617705092
Management Event@1617705112
Machine B Event@1617705132
Management Event@1617705132
Machine A Event@1617705135
Management Event@1617705152
Management Event@1617705172
Management Event@1617705192
Machine B Event@1617705200
Machine A Event@1617705211
Management Event@1617705212
Machine B Event@1617705219
Management Event@1617705232
Machine B Event@1617705241
Management Event@1617705252
Management Event@1617705272
Machine A Event@1617705289
Management Event@1617705292
Machine B Event@1617705310
Management Event@1617705312
Machine B Event@1617705321
Management Event@1617705332
Management Event@1617705352
Machine A Event@1617705356
Management Event@1617705372
Management Event@1617705392
Management Event@1617705412
Management Event@1617705432
Machine B Event@1617705440
Machine A Event@1617705451
Management Event@1617705452
Machine B Event@1617705454
Management Event@1617705472
Management Event@1617705492
Management Event@1617705512
Management Event@1617705532
Machine A Event@1617705533
Management Event@1617705552
Management Event@1617705572
Management Event@1617705592
Machine A Event@1617705602
Management Event@1617705612
Machine B Event@1617705613
Management Event@1617705632
Management Event@1617705652
Machine A Event@1617705654
Machine B Event@1617705658
Machine A Event@1617705662
Machine B Event@1617705663
Management Event@1617705672
Machine A Event@1617705677
Machine A Event@1617705680
Management Event@1617705692
Management Event@1617705712
Management Event@1617705732
Machine A Event@1617705736
Management Event@1617705752
Machine B Event@1617705761
Management Event@1617705772
Management Event@1617705792
Machine B Event@1617705797
Management Event@1617705812
Machine A Event@1617705831
Management Event@1617705832
Machine A Event@1617705836
Management Event@1617705852
Machine B Event@1617705871
Management Event@1617705872
Machine A Event@1617705884
Management Event@1617705892
Machine A Event@1617705892
Machine B Event@1617705894
Management Event@1617705912
Machine A Event@1617705919
Management Event@1617705932
Management Event@1617705952
Machine B Event@1617705964
Management Event@1617705972
Management Event@1617705992
Machine A Event@1617705995
Management Event@1617706012
Machine B Event@1617706020
Machine B Event@1617706026
Management Event@1617706032
End Event@1617703440
---------------------------
Final Stats:
Average Time of Production: 14s
---------------------------


Terminating Simulation!
```

Figure 1. Output als Image

Listing 11. Valgrind Output

```
==20065== HEAP SUMMARY:
==20065==     in use at exit: 0 bytes in 0 blocks
==20065==   total heap usage: 980 allocs, 980 frees, 106,848 bytes allocated
==20065==
==20065== All heap blocks were freed -- no leaks are possible
==20065==
```

3.2. Ausführen des Szenarios mit `step`

Es wird das ganze Szenario mithilfe von `step` ausgeführt. Als Abbruchbendinung wird die Menge der Produkte genutzt. Erwartet wird ein erfolgreicher Durchlauf.

Listing 12. Output

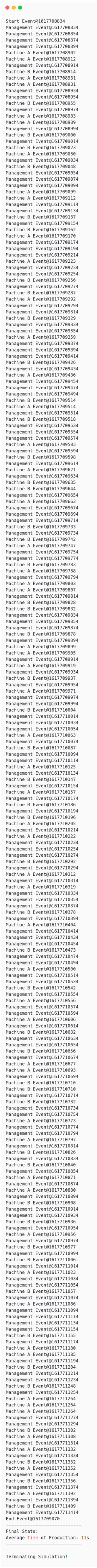
```
Start Event@1617708834
Management Event@1617708834
Management Event@1617708854
Management Event@1617708874
Management Event@1617708894
Machine A Event@1617708902
Machine A Event@1617708912
Management Event@1617708914
Machine B Event@1617708914
Machine A Event@1617708931
Machine A Event@1617708931
Management Event@1617708934
Management Event@1617708954
Machine B Event@1617708955
Management Event@1617708974
Machine A Event@1617708983
Machine A Event@1617708989
Management Event@1617708994
Machine B Event@1617709000
Management Event@1617709014
Machine B Event@1617709023
Machine A Event@1617709030
Management Event@1617709034
Machine B Event@1617709048
Management Event@1617709054
Management Event@1617709074
Management Event@1617709094
Machine A Event@1617709099
Machine A Event@1617709112
Management Event@1617709114
Management Event@1617709134
Machine B Event@1617709137
Management Event@1617709154
```

```
Machine B Event@1617709162
Machine A Event@1617709170
Management Event@1617709174
Management Event@1617709194
Management Event@1617709214
Machine A Event@1617709223
Management Event@1617709234
Management Event@1617709254
Machine B Event@1617709256
Management Event@1617709274
Machine B Event@1617709287
Machine A Event@1617709292
Management Event@1617709294
Management Event@1617709314
Machine B Event@1617709329
Management Event@1617709334
Management Event@1617709354
Machine A Event@1617709359
Management Event@1617709374
Management Event@1617709394
Management Event@1617709414
Machine B Event@1617709426
Management Event@1617709434
Machine A Event@1617709436
Management Event@1617709454
Management Event@1617709474
Management Event@1617709494
Machine B Event@1617709514
Machine A Event@1617709514
Management Event@1617709514
Machine B Event@1617709518
Management Event@1617709534
Management Event@1617709554
Management Event@1617709574
Machine A Event@1617709583
Management Event@1617709594
Machine B Event@1617709598
Management Event@1617709614
Machine A Event@1617709621
Management Event@1617709634
Machine B Event@1617709635
Machine B Event@1617709644
Management Event@1617709654
Machine A Event@1617709663
Management Event@1617709674
Management Event@1617709694
Management Event@1617709714
Machine B Event@1617709733
Management Event@1617709734
Machine B Event@1617709742
Machine A Event@1617709747
Management Event@1617709754
Management Event@1617709774
Machine B Event@1617709783
Machine A Event@1617709788
Management Event@1617709794
Machine A Event@1617709803
Machine A Event@1617709807
Management Event@1617709814
Machine B Event@1617709828
Machine B Event@1617709832
Management Event@1617709834
Management Event@1617709854
Management Event@1617709874
Machine B Event@1617709878
Management Event@1617709894
Machine A Event@1617709899
Machine A Event@1617709905
Management Event@1617709914
Machine A Event@1617709919
Management Event@1617709934
Machine B Event@1617709937
Management Event@1617709954
Machine A Event@1617709971
Management Event@1617709974
Management Event@1617709994
Machine B Event@1617710004
Management Event@1617710014
Management Event@1617710034
Management Event@1617710054
Machine A Event@1617710063
Management Event@1617710074
Machine B Event@1617710087
Management Event@1617710094
Management Event@1617710114
Machine A Event@1617710125
Management Event@1617710134
Machine B Event@1617710147
Management Event@1617710154
Machine A Event@1617710157
Management Event@1617710174
Machine B Event@1617710186
Management Event@1617710194
Machine B Event@1617710196
Machine A Event@1617710205
Management Event@1617710214
Machine A Event@1617710222
Management Event@1617710234
Management Event@1617710254
Management Event@1617710274
Machine B Event@1617710292
Management Event@1617710294
Machine A Event@1617710312
Management Event@1617710314
```

```
Machine A Event@1617710319
Management Event@1617710334
Management Event@1617710354
Management Event@1617710374
Machine B Event@1617710378
Management Event@1617710394
Machine A Event@1617710404
Management Event@1617710414
Management Event@1617710434
Management Event@1617710454
Machine B Event@1617710473
Management Event@1617710474
Management Event@1617710494
Machine A Event@1617710500
Management Event@1617710514
Management Event@1617710534
Machine B Event@1617710542
Management Event@1617710554
Machine A Event@1617710556
Management Event@1617710574
Management Event@1617710594
Machine A Event@1617710606
Management Event@1617710614
Machine B Event@1617710632
Management Event@1617710634
Management Event@1617710654
Machine B Event@1617710656
Management Event@1617710674
Machine A Event@1617710677
Machine A Event@1617710693
Management Event@1617710694
Machine B Event@1617710710
Machine B Event@1617710710
Management Event@1617710714
Machine B Event@1617710732
Management Event@1617710734
Management Event@1617710754
Machine A Event@1617710773
Management Event@1617710774
Management Event@1617710794
Machine A Event@1617710797
Management Event@1617710814
Machine B Event@1617710826
Management Event@1617710834
Machine B Event@1617710840
Management Event@1617710854
Machine A Event@1617710871
Management Event@1617710874
Machine A Event@1617710880
Management Event@1617710894
Machine B Event@1617710908
Management Event@1617710914
Management Event@1617710934
Machine B Event@1617710936
Management Event@1617710954
Machine A Event@1617710956
Management Event@1617710974
Machine B Event@1617710977
Management Event@1617710994
Machine B Event@1617711006
Management Event@1617711014
Machine A Event@1617711023
Management Event@1617711034
Management Event@1617711054
Machine B Event@1617711057
Management Event@1617711074
Machine A Event@1617711086
Management Event@1617711094
Management Event@1617711114
Management Event@1617711134
Management Event@1617711154
Machine B Event@1617711155
Management Event@1617711174
Machine A Event@1617711180
Machine A Event@1617711185
Management Event@1617711194
Machine B Event@1617711204
Management Event@1617711214
Management Event@1617711234
Machine B Event@1617711248
Management Event@1617711254
Machine A Event@1617711264
Machine A Event@1617711264
Machine A Event@1617711264
Management Event@1617711274
Management Event@1617711294
Machine B Event@1617711302
Machine A Event@1617711308
Management Event@1617711314
Machine A Event@1617711332
Management Event@1617711334
Machine B Event@1617711352
Machine A Event@1617711352
Management Event@1617711354
Machine B Event@1617711356
Management Event@1617711374
Machine A Event@1617711392
Management Event@1617711394
Machine B Event@1617711409
Management Event@1617711414
End Event@1617709070
----------------------------
Final Stats:
Average Time of Production: 11s
```

```
----------------------------

Terminating Simulation!
```



```
Start Event@1617708834
Management Event@1617708834
Management Event@1617708854
Management Event@1617708874
Management Event@1617708894
Machine A Event@1617708902
Machine A Event@1617708912
Management Event@1617708914
Machine B Event@1617708914
Machine A Event@1617708931
Machine A Event@1617708931
Management Event@1617708934
Management Event@1617708954
Machine B Event@1617708955
Management Event@1617708974
Machine A Event@1617708983
Machine A Event@1617708989
Management Event@1617708994
Machine B Event@1617709000
Management Event@1617709014
Machine B Event@1617709023
Machine A Event@1617709030
Management Event@1617709034
Machine B Event@1617709048
Management Event@1617709054
Management Event@1617709074
Management Event@1617709094
Machine A Event@1617709099
Machine A Event@1617709112
Management Event@1617709114
Management Event@1617709134
Machine B Event@1617709137
Management Event@1617709154
Machine B Event@1617709162
Machine A Event@1617709170
Management Event@1617709174
Management Event@1617709194
Management Event@1617709214
Machine A Event@1617709223
Management Event@1617709234
Management Event@1617709254
Machine B Event@1617709256
Management Event@1617709274
Machine B Event@1617709287
Machine A Event@1617709292
Management Event@1617709294
Management Event@1617709314
Machine B Event@1617709329
Management Event@1617709334
Management Event@1617709354
Machine A Event@1617709359
Management Event@1617709374
Management Event@1617709394
Management Event@1617709414
Machine B Event@1617709426
Management Event@1617709434
Machine A Event@1617709436
Management Event@1617709454
Management Event@1617709474
Management Event@1617709494
Machine B Event@1617709514
Machine A Event@1617709514
Management Event@1617709514
Machine B Event@1617709518
Management Event@1617709534
Management Event@1617709554
Management Event@1617709574
Machine A Event@1617709583
Management Event@1617709594
Machine B Event@1617709598
Management Event@1617709614
Machine A Event@1617709621
Management Event@1617709634
Machine B Event@1617709635
Machine B Event@1617709644
Management Event@1617709654
Machine A Event@1617709663
Management Event@1617709674
Management Event@1617709694
Management Event@1617709714
Machine B Event@1617709733
Management Event@1617709734
Machine B Event@1617709742
Machine A Event@1617709747
Management Event@1617709754
Management Event@1617709774
Machine B Event@1617709783
Machine A Event@1617709788
Management Event@1617709794
Machine A Event@1617709803
Machine A Event@1617709807
Management Event@1617709814
Machine B Event@1617709828
Machine B Event@1617709832
Management Event@1617709834
Management Event@1617709854
Management Event@1617709874
Machine B Event@1617709878
Management Event@1617709894
Machine A Event@1617709899
Machine A Event@1617709905
Management Event@1617709914
Machine A Event@1617709919
Management Event@1617709934
Machine B Event@1617709937
Management Event@1617709954
Machine A Event@1617709971
Management Event@1617709974
Management Event@1617709994
Machine B Event@1617710004
Management Event@1617710014
Management Event@1617710034
Management Event@1617710054
Machine A Event@1617710063
Management Event@1617710074
Machine B Event@1617710087
Management Event@1617710094
Management Event@1617710114
Machine A Event@1617710125
Management Event@1617710134
Machine B Event@1617710147
Management Event@1617710154
Machine A Event@1617710157
Management Event@1617710174
Machine B Event@1617710186
Management Event@1617710194
Machine B Event@1617710196
Machine A Event@1617710205
Management Event@1617710214
Machine A Event@1617710222
Management Event@1617710234
Management Event@1617710254
Management Event@1617710274
Machine B Event@1617710292
Management Event@1617710294
Machine A Event@1617710312
Management Event@1617710314
Machine A Event@1617710319
Management Event@1617710334
Management Event@1617710354
Management Event@1617710374
Machine B Event@1617710378
Management Event@1617710394
Machine A Event@1617710404
Management Event@1617710414
Management Event@1617710434
Management Event@1617710454
Machine B Event@1617710473
Management Event@1617710474
Management Event@1617710494
Machine A Event@1617710500
Management Event@1617710514
Management Event@1617710534
Machine B Event@1617710542
Management Event@1617710554
Machine A Event@1617710556
Management Event@1617710574
Management Event@1617710594
Machine A Event@1617710606
Management Event@1617710614
Machine B Event@1617710632
Management Event@1617710634
Management Event@1617710654
Machine B Event@1617710656
Management Event@1617710674
Machine A Event@1617710677
Machine A Event@1617710693
Management Event@1617710694
Machine B Event@1617710710
Machine B Event@1617710710
Management Event@1617710714
Machine B Event@1617710732
Management Event@1617710734
Management Event@1617710754
Machine A Event@1617710773
Management Event@1617710774
Management Event@1617710794
Machine A Event@1617710797
Management Event@1617710814
Machine B Event@1617710826
Management Event@1617710834
Machine B Event@1617710840
Management Event@1617710854
Machine A Event@1617710871
Management Event@1617710874
Machine A Event@1617710880
Management Event@1617710894
Machine B Event@1617710908
Management Event@1617710914
Management Event@1617710934
Machine B Event@1617710936
Management Event@1617710954
Machine A Event@1617710956
Management Event@1617710974
Machine B Event@1617710977
Management Event@1617710994
Machine B Event@1617711006
Management Event@1617711014
Machine A Event@1617711023
Management Event@1617711034
Management Event@1617711054
Machine B Event@1617711057
Management Event@1617711074
Machine A Event@1617711086
Management Event@1617711094
Management Event@1617711114
Management Event@1617711134
Management Event@1617711154
Machine B Event@1617711155
Management Event@1617711174
Machine A Event@1617711180
Machine A Event@1617711185
Management Event@1617711194
Machine B Event@1617711204
Management Event@1617711214
Management Event@1617711234
Machine B Event@1617711248
Management Event@1617711254
Machine A Event@1617711264
Machine A Event@1617711264
Machine A Event@1617711264
Management Event@1617711274
Management Event@1617711294
Machine B Event@1617711302
Machine A Event@1617711308
Management Event@1617711314
Machine A Event@1617711332
Management Event@1617711334
Machine B Event@1617711352
Machine A Event@1617711352
Management Event@1617711354
Machine B Event@1617711356
Management Event@1617711374
Machine A Event@1617711392
Management Event@1617711394
Machine B Event@1617711409
Management Event@1617711414
End Event@1617709070
---------------------------
Final Stats:
Average Time of Production: 11s
---------------------------


Terminating Simulation!
```

Figure 2. Output als Image

Listing 13. Valgrind Output

```
==21775==
==21775== HEAP SUMMARY:
==21775==     in use at exit: 0 bytes in 0 blocks
==21775==   total heap usage: 996 allocs, 996 frees, 107,333 bytes allocated
==21775==
==21775== All heap blocks were freed -- no leaks are possible
==21775==
```

3.3. Ausführen des Szenarios mit Zeit als Abbruchbedingung

Es wird als Abbruchbedingung eine Minute gewählt. Erwartet wird ein Abbruch nach einer Minute.

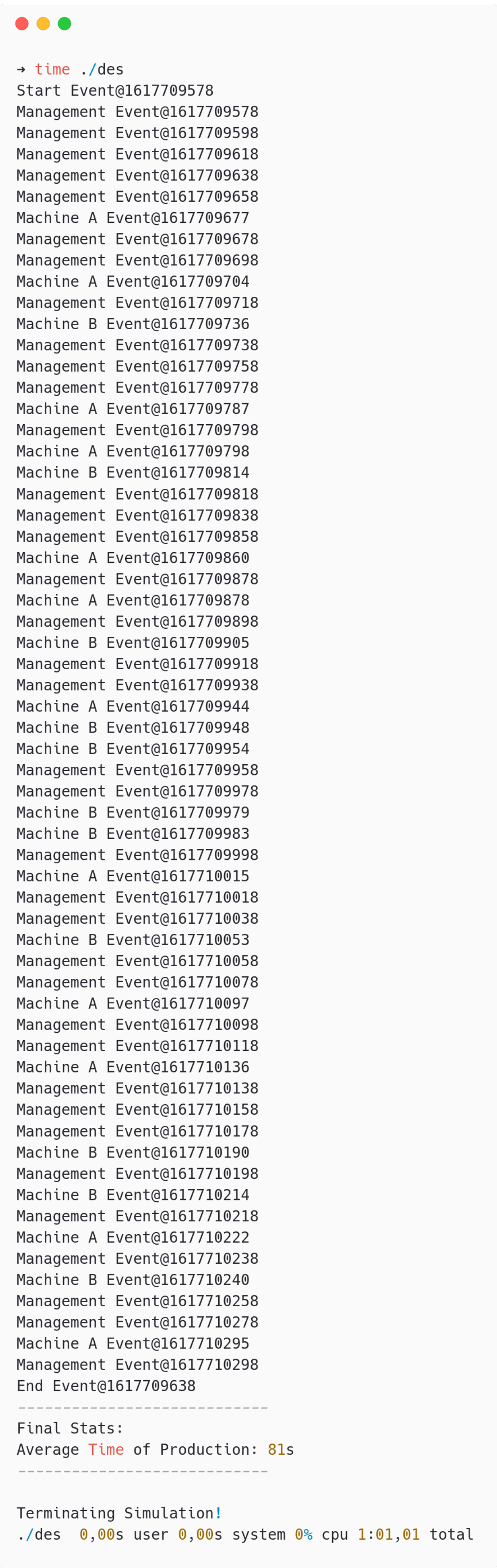
Man sieht am Output, dass nach einer Minute und einer Sekunde abgebrochen worden ist (Letzte Zeile: `./des 0,00s user 0,00s system 0% cpu 1:01,01 total`). Zum Messen der Zeit wurde das Time Command von Linux genutzt.

Listing 14. Output

```
▯  time ./des
Start Event@1617709578
Management Event@1617709578
Management Event@1617709598
Management Event@1617709618
Management Event@1617709638
Management Event@1617709658
Machine A Event@1617709677
Management Event@1617709678
Management Event@1617709698
Machine A Event@1617709704
Management Event@1617709718
Machine B Event@1617709736
Management Event@1617709738
Management Event@1617709758
Management Event@1617709778
Machine A Event@1617709787
Management Event@1617709798
Machine A Event@1617709798
Machine B Event@1617709814
Management Event@1617709818
Management Event@1617709838
Management Event@1617709858
Machine A Event@1617709860
Management Event@1617709878
Machine A Event@1617709878
Management Event@1617709898
```

```
Machine B Event@1617709905
Management Event@1617709918
Management Event@1617709938
Machine A Event@1617709944
Machine B Event@1617709948
Machine B Event@1617709954
Management Event@1617709958
Management Event@1617709978
Machine B Event@1617709979
Machine B Event@1617709983
Management Event@1617709998
Machine A Event@1617710015
Management Event@1617710018
Management Event@1617710038
Machine B Event@1617710053
Management Event@1617710058
Management Event@1617710078
Machine A Event@1617710097
Management Event@1617710098
Management Event@1617710118
Machine A Event@1617710136
Management Event@1617710138
Management Event@1617710158
Management Event@1617710178
Machine B Event@1617710190
Management Event@1617710198
Machine B Event@1617710214
Management Event@1617710218
Machine A Event@1617710222
Management Event@1617710238
Machine B Event@1617710240
Management Event@1617710258
Management Event@1617710278
Machine A Event@1617710295
Management Event@1617710298
End Event@1617709638
----------------------------
Final Stats:
Average Time of Production: 81s
----------------------------

Terminating Simulation!
./des  0,00s user 0,00s system 0% cpu 1:01,01 total
```



```
➜ time ./des
Start Event@1617709578
Management Event@1617709578
Management Event@1617709598
Management Event@1617709618
Management Event@1617709638
Management Event@1617709658
Machine A Event@1617709677
Management Event@1617709678
Management Event@1617709698
Machine A Event@1617709704
Management Event@1617709718
Machine B Event@1617709736
Management Event@1617709738
Management Event@1617709758
Management Event@1617709778
Machine A Event@1617709787
Management Event@1617709798
Machine A Event@1617709798
Machine B Event@1617709814
Management Event@1617709818
Management Event@1617709838
Management Event@1617709858
Machine A Event@1617709860
Management Event@1617709878
Machine A Event@1617709878
Management Event@1617709898
Machine B Event@1617709905
Management Event@1617709918
Management Event@1617709938
Machine A Event@1617709944
Machine B Event@1617709948
Machine B Event@1617709954
Management Event@1617709958
Management Event@1617709978
Machine B Event@1617709979
Machine B Event@1617709983
Management Event@1617709998
Machine A Event@1617710015
Management Event@1617710018
Management Event@1617710038
Machine B Event@1617710053
Management Event@1617710058
Management Event@1617710078
Machine A Event@1617710097
Management Event@1617710098
Management Event@1617710118
Machine A Event@1617710136
Management Event@1617710138
Management Event@1617710158
Management Event@1617710178
Machine B Event@1617710190
Management Event@1617710198
Machine B Event@1617710214
Management Event@1617710218
Machine A Event@1617710222
Management Event@1617710238
Machine B Event@1617710240
Management Event@1617710258
Management Event@1617710278
Machine A Event@1617710295
Management Event@1617710298
End Event@1617709638
----------------------------
Final Stats:
Average Time of Production: 81s
----------------------------

Terminating Simulation!
./des  0,00s user 0,00s system 0% cpu 1:01,01 total
```

Figure 3. Output als Image

Listing 15. Valgrind Output

```
==23928==
==23928== HEAP SUMMARY:
==23928==     in use at exit: 0 bytes in 0 blocks
==23928==   total heap usage: 282 allocs, 282 frees, 83,824 bytes allocated
==23928==
==23928== All heap blocks were freed -- no leaks are possible
==23928==
==23928== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

3.4. Ausführen des Szenarios mit Zeit als Abbruchbedingung mit `step`

Es wird als Abbruchbedingung eine Minute gewählt, es wird mit `step` ausgeführt. Erwartet wird ein Abbruch nach einer Minute.

Man sieht am Output, dass nach einer Minute und einer Sekunde abgebrochen worden ist. Zum Messen der Zeit wurde das Time Command von Linux genutzt.

Listing 16. Output

```
□  time ./des
Start Event@1617710059
Management Event@1617710059
Management Event@1617710079
Management Event@1617710099
Machine A Event@1617710111
Management Event@1617710119
Machine A Event@1617710135
Management Event@1617710139
Management Event@1617710159
Machine B Event@1617710160
Management Event@1617710179
Machine A Event@1617710181
Management Event@1617710199
Machine A Event@1617710208
Management Event@1617710219
Management Event@1617710239
Machine B Event@1617710243
Management Event@1617710259
Machine A Event@1617710262
Machine B Event@1617710277
Management Event@1617710279
Machine B Event@1617710281
Management Event@1617710299
Machine A Event@1617710310
Management Event@1617710319
```

```
Machine B Event@1617710328
Management Event@1617710339
Management Event@1617710359
Machine B Event@1617710371
Machine A Event@1617710379
Management Event@1617710379
Management Event@1617710399
Machine A Event@1617710400
Management Event@1617710419
Management Event@1617710439
Machine B Event@1617710450
Management Event@1617710459
Machine A Event@1617710462
Management Event@1617710479
Management Event@1617710499
Management Event@1617710519
Machine B Event@1617710521
Machine B Event@1617710537
Management Event@1617710539
Management Event@1617710559
Machine A Event@1617710560
Management Event@1617710579
Management Event@1617710599
Management Event@1617710619
Machine A Event@1617710624
Management Event@1617710639
Management Event@1617710659
Machine B Event@1617710665
Management Event@1617710679
Machine B Event@1617710691
Management Event@1617710699
Management Event@1617710719
Machine A Event@1617710723
Management Event@1617710739
Machine B Event@1617710745
Machine A Event@1617710752
Management Event@1617710759
End Event@1617710120
----------------------------
Final Stats:
Average Time of Production: 76s
----------------------------

Terminating Simulation!
./des  0,00s user 0,00s system 0% cpu 1:02,01 total
```

Output als Image

image::./img/testcase_4.png

Listing 17. Valgrind Output

```
==25449==
==25449== HEAP SUMMARY:
==25449==     in use at exit: 0 bytes in 0 blocks
==25449==   total heap usage: 278 allocs, 278 frees, 83,689 bytes allocated
==25449==
==25449== All heap blocks were freed -- no leaks are possible
==25449==
==25449== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```