

Øvelse 0: Bli kjent med Raspberry Pi

1) Start maskinen

2) Logg inn på maskinen (bruker: `pi`, passord: `raspberrypi`)

3) Se på alle filene i nåværende katalog

```
pwd  
ls -al
```

4) Skift ned til underkatalogen `python_games`, og list alle filene som slutter på `.py`

```
cd python_games  
pwd  
ls -al  
ls *.py
```

5) Start spillet Wormy

```
python wormy.py
```

6) Gå tilbake til din hjemmekatalog

```
cd  
pwd  
ls -al
```

Øvelse 1: Hello (obligatorisk)

Logg inn på maskinen (pi/raspberry).

Gå til hjemmeområdet ditt:

```
cd
```

Lag en ny programfil med en teksteditor:

```
nano hello.cpp
```

og så skriver du inn programmet som du ser til høyre.

Når du er ferdig med å skrive inn programmet kompilerer du det med en C++ kompilator:

```
g++ -o hello hello.cpp
```

og så kjører du programmet slik:

```
./hello
```

```
#include <iostream>

int main()
{
    std::cout << "Hello!" << std::endl;
    return 0;
}
```

Øvelse 2: Hello, del 2

Logg inn på maskinen (pi/raspberry).

Gå til hjemmeområdet ditt:

```
cd
```

Lag en kopi av den gamle filen, og rediger kopien med en teksteditor:

```
cp hello.cpp hello2.cpp  
nano hello2.cpp
```

og så skriver du inn programmet som du ser nedenfor.

Kompiler og kjør:

```
g++ -o hello2 hello2.cpp  
./hello2
```

```
#include <iostream>  
  
int main()  
{  
    std::string name;  
    std::cout << "What is your name? " << std::flush;  
    std::cin >> name;  
    int i = 0;  
    while (i < 10) {  
        std::cout << "Hello " << name << "!" << std::endl;  
        i = i + 1;  
    }  
    return 0;  
}
```

Øvelse 3: MyPaint, del I

Dette er et lite program som tegner to streker på skjermen.

Gå til hjemmeområdet ditt:

```
cd
```

Lag en ny programfil med en teksteditor:

```
nano mypaint1.cpp
```

og så skriver du inn programmet, kompilerer og kjører:

```
g++ -lncurses -o mypaint1 mypaint1.cpp  
./mypaint1
```

Klarer du å tegne noe som ligner på et hus?

```
#include <ncurses.h>  
  
int main()  
{  
    initscr();  
    noecho();  
    curs_set(0);  
  
    mvprintw(0,10,"My Paint");  
  
    int r = 10;  
    int c = 12;  
  
    while (c < 30) {  
        mvaddch(r,c,'X');  
        c++;  
    }  
  
    while (r < 20) {  
        mvaddch(r,c,'X');  
        c--;  
        r++;  
    }  
  
    getch();  
  
    endwin();  
  
    return 0;  
}
```

Øvelse 4: MyPaint, del 2

Gjenbruk koden fra del 1:

```
cp mypaint1.cpp mypaint2.cpp  
nano mypaint2.cpp
```

Endre slik at koden blir som den til høyre.

Kompiler og kjør:

```
g++ -o mypaint2 mypaint2.cpp  
./mypaint2
```

```
#include <ncurses.h>  
  
int main()  
{  
    initscr();  
    noecho();  
    curs_set(0);  
  
    mvprintw(0,10,"My Paint");  
  
    int r = 10;  
    int c = 12;  
    bool quit = false;  
    while (!quit) {  
        mvaddch(r,c,'X');  
        int ch = getch();  
        if (ch == 'a')  
            c--;  
        else if (ch == 'd')  
            c++;  
        else if (ch == 'w')  
            r--;  
        else if (ch == 's')  
            r++;  
        else if (ch == 'q')  
            quit = true;  
    }  
  
    endwin();  
  
    return 0;  
}
```

Øvelse 5: MyPaint, del 3

Gjenbruk gjerne programmet fra del2.

Skriv inn og kjør programmet til høyre. Dette er et interaktivt program hvor du styrer "musen" med `a`, `d`, `w`, `s` og så setter du ned eller løfter pennen med `p`. `q` for å avslutte.

Legg merke til hvordan vi bruke switch isteden for mange if-else. Hva skjer hvis du glemmer å skrive break, feks etter case 'd'?

mypaint3.cpp

```
#include <ncurses.h>

int main()
{
    initscr();
    noecho();
    curs_set(0);

    mvprintw(0,10,"My Paint");

    int r = 10;
    int c = 12;
    bool paint = false;
    bool quit = false;
    while (!quit) {
        mvaddch(r,c,'X');
        int ch = getch();
        if (!paint)
            mvaddch(r,c,' ');
        switch (ch) {
            case 'a': c--; break;
            case 'd': c++; break;
            case 'w': r--; break;
            case 's': r++; break;
            case 'q': quit = true; break;
            case 'p': paint = !paint; break;
        }
    }

    endwin();

    return 0;
}
```

Øvelse 6: Calc, del I

Alle større programmer består av flere filer som oversettes til maskinspråk hver for seg (slik at vi får en .o objektfil) og så lenkes disse objektfilene sammen med en såkalt linker slik at vi får et kjørbart program. Skriv inn filene under, og så kjører du disse kommandoene:

```
g++ -c my_calculator.cpp
g++ -c calc1.cpp
g++ -o calc1 my_calculator.o calc1.o
./calc1
```

Hva er svaret?

my_calculator.hpp

```
#include <stack>

class my_calculator {
public:
    void push(double value);
    double pop();
    double top() const;
    void add();
    void sub();
private:
    std::stack<double> stack;
};
```

calc1.cpp

```
#include "my_calculator.hpp"
#include <iostream>

int main()
{
    my_calculator calc;

    calc.push(35);
    calc.push(9);
    calc.add();
    calc.push(2);
    calc.sub();
    std::cout << calc.top() << std::endl;
}
```

my_calculator.cpp

```
#include "my_calculator.hpp"
#include <stdexcept>

void my_calculator::push(double value)
{
    stack.push(value);
}

double my_calculator::top() const
{
    if (stack.empty())
        throw std::runtime_error("stack error");
    return stack.top();
}

double my_calculator::pop()
{
    double value = top();
    stack.pop();
    return value;
}

void my_calculator::add()
{
    push( pop() + pop() );
}

void my_calculator::sub()
{
    double op1 = pop();
    double op2 = pop();
    push( op2 - op1 );
}
```


Øvelse 7: Calc, del 2

Nå skal vi lage et program med et brukergrensesnitt ved å gjenbruke modulen som vi laget i del 1.

Skriv inn filen til høyre og link med objektfilen my_calculator.o:

```
g++ -c my_calculator.cpp
g++ -c calc2.cpp
g++ -o calc2 my_calculator.o calc2.o
./calc2
```

Prøv å forbedre programmet slik at det også håndterer multiplikasjon og divisjon. Da blir du nok nødt til å endre alle tre filene.

calc2.cpp

```
#include "my_calculator.hpp"

#include <iostream>
#include <sstream>
#include <stdexcept>
#include <stdlib.h>

double to_double(const std::string & str)
{
    std::istringstream ss(str);
    double d;
    ss >> d;
    if ( !(ss >> std::ws).eof() )
        throw std::invalid_argument("huh?");
    return d;
}

int main()
{
    my_calculator calc;
    std::string input;
    while (std::cin >> input) {
        try {
            if (input == "q")
                exit(0);
            if (input == "p")
                std::cout << calc.top() << std::endl;
            else if (input == "+")
                calc.add();
            else if (input == "-")
                calc.sub();
            else
                calc.push(to_double(input));
        } catch(std::exception & ex) {
            std::cout << ex.what() << std::endl;
        }
    }
    return 0;
}
```

