TDT4102 – Aug 2019, utvidet løsningsforslag.

All kode er testet og kjørt. Dokumentet viser i hovedsak én av mange mulige løsninger som ville ha gitt maksimal uttelling på eksamen. Vi ber om forståelse for at vi ikke har kapasitet til å lage et LF som viser mange ulike løsningsmuligheter på de ulike delspørsmål. (Husk at alle ikke-trivielle programmeringsoppgaver kan løses på utallige måter.) I dette *utvidete* LF viser vi alternative løsninger på noen få deloppgaver.

Sist oppdatert: 26/8-2019, *Lasse Natvig*

Generell kommentar: I de fleste av deloppgavene så har detaljer som linjeskift, komma osv. svært liten betydning for om studenten klarer å vise at han/hun behersker det vi spør om, og da trekkes det ikke for slike detaljer. Vi trekker heller ikke for mindre syntaks-feil som en kompilator raskt ville ha hjulpet deg med.

Oppgave 1: Kodeforståelse
1a) 20 42.42 41
1b) 2 3 5 8 13
1c) iik
1d) 42 33 2!
1e) A AB ABA ABAAB
1f) trololo
1g) nam pngm catz
1h) SUSS

Oppgave 2:
2a)
```
double sum(vector<double>& x) {
	double sum = 0;
	for (double xi : x) {
		sum += xi;
	}
	return sum;
}
```
Alternativ løsning med bruk av accumulate fra <algorithm> (Se PPP side 771)
```
double sum(vector<double>& x) {
	return accumulate(x.begin(), x.end(), 0.0);
}
```
2b)
```
double mean(vector<double>& x) {
	return sum(x) / x.size();
}
```
2c)
```
void read_csv(string filename, vector<double>& x, vector<double>& y) {
	double xi;
	double yi;
	ifstream file{ filename };
	if (!file) {
		throw "Couldn't read file " + filename;
	}
	while (file >> xi >> yi) {
		x.push_back(xi);
		y.push_back(yi);
	}
}
```

2d)

```
pair<double, double> linreg(vector<double>& x, vector<double>& y) {
	double xmean = mean(x);
	double ymean = mean(y);
	double cov = 0.0;
	double var = 0.0;
	for (unsigned int i = 0; i < x.size(); i++) {
		var += pow(x[i] - xmean, 2);
		cov += (x[i] - xmean) * (y[i] - ymean);
	} // var is now in fact = var*n, and cov is cov*n. No need to divide
	  // by n since they are used only in a = cov/var below
	double a = cov / var;
	double b = ymean - a * xmean;
	return { a, b };
}
```

2e)

```
vector<double> linpred(vector<double> &x, double a, double b) {
	vector<double> y;
	for (auto xi : x) {
		y.push_back(a * xi + b);
	}
	return y;
}
```

2f)

```
double r2(vector<double>& y, vector<double>& y_pred) {
	double ymean = mean(y);
	double sstot = 0.0;
	double ssres = 0.0;
	for (unsigned int i = 0; i < y.size(); i++) {
		sstot += pow(y[i] - ymean, 2);
		ssres += pow(y[i] - y_pred[i], 2);
	}
	return 1 - (ssres / sstot);
}
```

Kommentar: Det er ikke eksplisitt oppgitt i oppgaveteksten at R^2 skal returneres, men det bør fremgå klart av konteksten siden returverdi double er gitt og funksjonen heter r2.

2g)

```
int mainOppgave2g() {
	vector<double> x;
	vector<double> y;
	read_csv("data.csv", x, y);
	auto p = linreg(x, y);
	double a = p.first;
	double b = p.second;
	cout << "Linreg: y = ax + b" << endl;
	cout << "a = " << a << endl;
	cout << "b = " << b << endl;
	vector<double> y_pred = linpred(x, a, b);
	double R2 = r2(y, y_pred);
	cout << "R^2 = " << R2 << endl;
	return 0;
}
```

Kommentar: En trenger ikke a= …, b = … osv. i utskrift for full score, da oppgaveteksten ikke spesifiserer dette

Oppgave 3:

3a)

```
Location::Location(string str, Point pt)
	: name{ str }, p{ pt } {}
```

3b)

```
class BikeStation {
	Location loc;
	unsigned int capacity = 0;
	unsigned int bikes = 0;
	Vector_ref<Shape> display;
public:
	unsigned int getBikes() const;
	void setBikes(unsigned int b);
	string getName() { return loc.name; }
};
```

3c)

```
void BikeStation::setBikes(unsigned int b) {
	bikes = b;
}
unsigned int BikeStation::getBikes() const {
	return bikes;
}
```

3d)

```
string BikeStation::status() {
	return to_string(bikes) + " out of " + to_string(capacity);
}
```

3e)

```
BikeStation::BikeStation(Location where, unsigned int cap, unsigned int numBikes)
	: loc{ where }, capacity{ cap }, bikes{ numBikes } {
	// Punkt 1)
	Rectangle* r = new Rectangle{ loc.p, dispWidth, dispHeigth };
	r->set_fill_color(Color::white);
	display.push_back(r);
	// Punkt 2), for alternativ kode etter "the Bjarne way", se punkt 3
	Text* t = new Text{ loc.p, loc.name };
	t->set_color(Color::blue);
	t->set_font_size(20);
	display.push_back(t);
	// Punkt 3), "The Bjarne way"
	display.push_back(new Text{ Point{loc.p.x + 2, loc.p.y + 15}, status() });
	display[display.size() - 1].set_color(Color::black);
}
```

3f)

```
map<string, int> simulateOneDay(vector<BikeStation*> allBikes) {    map<string, int>
unsuccessfullRides;
	const int numStations = allBikes.size();
	int from = 0;
	int to = 0;
	for (int i = 0; i < ridesPerDay; i++) {
		from = rand() % numStations;
		to = rand() % numStations;
		while (to == from) { // maybe not the best performance for this, but simple
			to = rand() % numStations;
		}
		int avail = allBikes[from]->getBikes();
		int bikesThere = allBikes[to]->getBikes();
		int space = allBikes[to]->getCapacity() - bikesThere;
		if ((avail > 0) && (space > 0)) {
			allBikes[from]->setBikes(--avail);
			allBikes[to]->setBikes(++bikesThere);
		}
		else {
			unsuccessfullRides[allBikes[from]->getName()]++;
		}
	}
	return unsuccessfullRides;
}
```

3g)

```
void printStats(map<string, int> result) {
	cout << "Unsuccessfull rides:\n";
	for (const auto res : result) {
		cout << res.second << " bike trips refused at " << res.first << endl;
	}
	cout << endl;
}
```

Oppgave 4 : Ringbuffer

4a)

```
RingBuf::RingBuf(int capacity)
	: buf{ new char[capacity] }, capacity{ capacity }, start{ 0 }, size{ 0 } { }
```

4b)

```
RingBuf::RingBuf(const RingBuf &other) : buf{ new char[other.capacity] },
		capacity{ other.capacity }, start{ other.start }, size{ other.size } {
	for (int i = 0; i < capacity; i++) {
		buf[i] = other.buf[i];
	}
}
```

Kommentar: Forelesning bruker for-løkke som ovenfor men boka bruker std:copy() (se side 635)

```
// for (int i = start; i < start + size; i++) {
//     buf[i % capacity] = other.buf[i % capacity]; } er også fullgod løsning og gir full score
```

4c)

```
RingBuf::~RingBuf() { delete[] buf; }
```

4d)

```
RingBuf::RingBuf(RingBuf &&other)
	: buf{ other.buf }, capacity{ other.capacity }, start{other.start}, size{other.size} {
	other.buf = nullptr;
	other.capacity = 0;
	other.start = 0;
	other.size = 0;
}
```

4e)

```
RingBuf& RingBuf::operator=(RingBuf rhs) {
	char* b = new char[rhs.capacity];
	for (int i = 0; i < rhs.capacity; i++) {
		b[i] = rhs.buf[i];
	}
	delete[] buf;
	capacity = rhs.capacity;
	start = rhs.start;
	size = rhs.size;
	buf = b;
	return *this;
}
```

Kommentar: Dette er lærebokas metode som også er forelest, men boka bruker std:copy() (PPP side 635). Riktig bruk av «copy&swap»-teknikken ga også full score: `RingBuf& RingBuf::operator=(RingBuf rhs) {`

```
	swap(buf, rhs.buf);
	swap(size, rhs.size);
	swap(start, rhs.start);
	swap(end, rhs.end);
	return *this;
}
```

4f)

```
void RingBuf::write(char c) {
	if (size == capacity) { // Overskriver starten
		buf[start++] = c;
		if (start == capacity) {
			start = 0;
		}
	}
	else {
		buf[(start + size) % capacity] = c;
		size++;
	}
}
```

4g)

```
char RingBuf::read() {
	if (size == 0) {
		throw "Attempting to read an empty buffer!";
	}
	char c = buf[start++];
	if (start == capacity) {
		start = 0;
	}
	size--;
	return c;
}
```

4h)

```
void RingBuf::write(string s) {
	for (char c : s) {
		write(c);
	}
}
```

eller `void RingBuf::write(string s) {`

```
	for (unsigned int i = 0; i < s.length(); i++) {
		write(s[i]);
	}
}
```

4i)

```
string RingBuf::read(int count) {
	if (count > size || count == -1) {
		count = size;
	}
	string s;
	for (int i = 0; i < count; i++) {
		s += read();
	}
	return s;
```

}Kommentar: I oppgaveteksten sto det "Hvis count er større enn *capacity* eller count er -1..."
Det var en trykkfeil, det skulle stått "Hvis count er større enn *size*...". En student oppdaget at dette var ulogisk og kommenterte det i besvarelsen. Begge løsninger ble godtatt ved retting.

4j)

```
string RingBuf::peek() {
	string s;
	for (int i = start; i < start + size; i++) {
		s += buf[i % capacity];
	}
	return s;
}
```

4k)

```
void testRingBuf() {
	RingBuf rb = RingBuf(5);
	string s;

	assert(rb.start == 0);
	assert(rb.size == 0);
	assert(rb.capacity == 5);
	assert(rb.peek() == "");

	rb.write("ABC");
	assert(rb.start == 0);
	assert(rb.size == 3);
	assert(rb.capacity == 5);
	assert(rb.peek() == "ABC");

	rb.write("DEF");
	assert(rb.start == 1);
	assert(rb.size == 5);
	assert(rb.capacity == 5);
	assert(rb.peek() == "BCDEF");

	s = rb.read(3);
	assert(s == "BCD");
	assert(rb.start == 4);
	assert(rb.size == 2);
	assert(rb.capacity == 5);
	assert(rb.peek() == "EF");

	s = rb.read(-1);
	assert(s == "EF");
	assert(rb.start == 1);
	assert(rb.size == 0);
	assert(rb.capacity == 5);
	assert(rb.peek() == "");
}
```

--oooOooo---...