5. labor kis ZH, 2015. április 30., csütörtök 10-12

NÉV (olvasható): NEPTUN kód:

max. 10p, megoldási idő: 15 perc

1. Adott a következő osztály:

```
class Kutya{
    double suly;
public:
    Kutya(double suly) :suly(suly){}
    void Print()const{ std::cout << "Suly: " << suly << std::endl; }
};
```

Hozz létre ebből örökléssel Bernat osztályt, melynek privát tagváltozója a kutya nyakában lévő hordóban lévő rum mennyisége. Készíts konstruktort, mellyel minden változót (az ősosztályban lévőt is) beállítasz, és készíts Print() függvényt, mellyel kiírod az összes tagváltozó értékét (az ősoben lévőt is). A Kutya osztályt nem módosíthatod.(4p)

```
class Bernat :public Kutya{
    double rum;
public:
    Bernat(double suly, double rum) :Kutya(suly), rum(rum){}
    void Print()const{
        Kutya::Print();
        std::cout << "Rum: " << rum << std::endl;
    }
};
```

2. Készíts generikus függvényt, amely egy tömböt kap paraméterként, és megkeresi a tömb maximális értékű elemét, melynek indexét adja vissza! A generikus változó a tárolt elemek típusa. Készíts specializációt a függvényhez const char* pointereket tároló tömb esetére, mely a mutatott sztringek között keresse a „legnagyobbat”. Felhasználhatod a cstring vagy string.h strcmp függvényét is. Írj kódrészletet, amelyben meghívod a maximumkereső függvényt (egy tömbbel, a típusát szabadon megválaszthatod). (6p)

```
template <typename T>
int max(T *t, int n){
    int m = 0;
    for (int i = 1; i < n; i++)
        if (t[i]>t[m])
            m = i;
    return m;
}

template<>
int max(const char **t, int n){
    int m = 0;
    for (int i = 1; i < n; i++)
        if (strcmp(t[i],t[m])>0)
            m = i;
    return m;
}

int t[6] = { 2, 8, 4, 5, 9, 1 };
int m = max(t, 6);
```