

Nimi: \_\_\_\_\_

Opiskelijanumero: \_\_\_\_\_

					=	
--	--	--	--	--	---	--

--

Maximipisteet 30: Asteikko: 27-30p: 5, 24-26p: 4, 21-23p: 3, 18-20p: 2, 15-17p: 1

Kysymyksen 1 vastataan rastittamalla tällä sivulla  
Kysymyksiin 2-5 vastataan erillisellä paperilla

Apuvälineitä ei sallittu

I. Monivalinta; valitse korkeintaan yksi vaihtoehto (6p)

- I. `#define MAX(a,b) ((a>b)?(a):(b))`  
`MAX(3+9, 11)`  
Millä ylläolevat rivit korvataan C/C++ esikäntäjässä
- 12
  - 3+9
  - `((3+9>11)?(3+9):(11))`
  - `((12>11)?12:11)`
- II. `int var1;`  
Jos ylläoleva muuttuja on on määritelty tiedostotasolla (siis funktioiden ulopuolella), onko muuttuja käytettävissä toisessa c/c++-tiedostossa?
- Kyllä, käyttämällä `register`-avainsanaa
  - Ei, olisi alunperin pitänyt määritellä `global`-tyyppiseksi
  - Kyllä, käyttämällä avainsanaa `extern`
  - Ei, olisi alunperin pitänyt määritellä `static`-tyyppiseksi
- III. Mikä seuraavista funktioprototyypeistä kannatta käyttää (C++), jos `MyFunc` käyttää `QPoint`-tietueen dataa, mutta ei muuta tietueen dataa
- `void MyFunc(QPoint)`
  - `void MyFunc(QPoint *)`
  - `void MyFunc(Qpoint const &)`
  - `void MyFunc(Qpoint &)`
- IV. `struct customer **ptr = malloc(sizeof(struct customer))`  
Annettu ylläoleva koodirivi, mikä seuraavista voidaan käyttää osoittimen reallokaatioon 10:lle elementille
- `ptr += malloc( 9 * sizeof( struct customer ) );`
  - `ptr = realloc( ptr, 10 * sizeof( struct customer));`
  - `ptr = realloc( ptr, 9 * sizeof( struct customer ) );`
  - `realloc( ptr, 9 * sizeof( struct customer ) );`
- V. `char txt[10] = "Hello world!\0";`  
Montako tavua varataan ylläolevassa koodissa?
- 10
  - 11
  - 12
  - 13
- VI. Seuraavat koodirivit käännetään C-kääntäjässä:  
`#include <malloc.h>`  
`char *myString[] = malloc(sizeof(char)+12) + 5;`  
Mikä seuraavista pitää paikkansa:
- Käännös epäonnistuu virhetilanteen takia
  - Käännös epäonnistuu; syntaksi väärä
  - Käännös jatkuu varoitusilmoituksella
  - Käännös keskeytyy toiseen riviin

## 2. Tietyssä koneessa, tietyllä kääntäjällä pätee

```
sizeof(char) = 1
sizeof(int) = 4
sizeof(char *) = sizeof(int *) = sizeof(void *) = 8
sizeof(float) = 4
sizeof(double) = 8
```

Paljonko muistia varataan (bytes) ja mikä/mitkä datatyypit ovat kyseessä alla olevien rivien kohdalla (6p)

```
I.   char *a[] = {NULL, NULL, NULL, (char*)0x10};
II.  void **b[10];
III. char *c[16];
IV.  int d[30];
V.   double e[12], *f[12];
VI.  int d[] = {0.12, 3, 5};
```

## 3. Usein C-lib-kirjastoissa implementoidaan funktio nimeltään `_strdup()` (ej ANSI-standardissa, mutta löytyy esim. Microsoft C/C++)

```
char *_strdup(const char *string);
```

Funktio käyttää dynaamista muistiallokointia (`malloc()`) uuden muistialueen varaamiseen, jonka jälkeen merkkijono kopioidaan varattuun muistiin. Paluuarvo on osoitin uuteen kopioon, tai `NULL` jos muista ei voitu alokoida. Implementoi `_strdup()` (6p)

## 4. Kirjoita C-kielessä tarvittavat funktiot pino-toiminnallisuutta varten allaolevien prototyyppien mukaisesti. Pino pitää pystyä varastoida integer-tyyppistä dataa. Deklaroi myös tarvittava `STACK`-tietuetta (3p)

```
STACK *initstack(unsigned size) /* Initialisoi uusi pino,
                                jossa maks. size elementtiä */
                                stack med max size element*/
void push(int el);               /* Lisää elementti pinoon */
int pop();                       /* Hae elementti pinosta */
```

b) Implementoi vastaavaa toiminnallisuutta, mutta tällä kertaa hyödyntämällä C++-kielen ominaisuuksia (3p).

## 5. Implementoi C++ avulla yksinkertainen versio luokasta `BigInteger` (lyhyesti `BI`), jolla voidaan tallentaa mielivaltaisen pitkiä kokonaislukuja. Luokkaimplementaatio tulee tukea seuraavaa main-funktiota:

```
int main() {
    BI b1(12345, 20); // value 12345, 20 digits
    BI b2(98765); // value 98765, 100 digits
    BI b3; // value 0, 100 digits
    BI b4("9999999999", 20); // value 9999999999, 20 digits
    BI b5("111111111111"); // value 111111111111, 100 digits
    BI b6 = b1; // same value and same number of digits as b1
    cin >> b3;
    b3 += b5;
    b1 = b2 + b3 + b5;
    cout << b3 << " " << b1 << endl;
}
```

Voidaan oletta 1) kaikki numerot ovat positiivisia, 2) muodostinfunktioiden argumentit ovat oikein annettu, 3) ainoastaan numroita argumenteissa (ei white-space). Implementoi ainoastaan tarvittava toiminnallisuus, jotta yllä oleva koodi voidaan ajaa. Koodi saa käyttää `<string>` ja `<iostream>`, mutta ei muita kirjastoja. (6p)