Rispondere alle domande a risposta multipla annerendo la casella corrispondente alla risposta corretta. Ogni domanda ha una ed una sola risposta corretta.

Cognome e Nome: .....

Matricola: .....

Domanda 1 Un garbage collector:

- [x] Può essere implementato tramite la tecnica detta "mark and sweep", che riesce sempre ad identificare tutta la memoria allocata dinamicamente ma non più utilizzata
- [ ] Richiede un'implementazione complessa, usando la tecnica dei *tombstone* (pietre tombali)
- [ ] E' implementabile solo in linguaggi di programmazione funzionali
- [ ] E' implementabile tramite la tecnica di lucchetti e chiavi, che però può causare dei memory leak
- [ ] Nessuna delle altre risposte

Domanda 2 Se gli array sono memorizzati *per colonne* ed `short int a[100][100]` è un array multidimensionale di interi corti (si assuma che la dimensione di uno `short int` sia 2 byte) con `a[0][0]` che ha indirizzo `0x4100`, qual'è l'indirizzo di `a[5][10]`?:

- [ ] 0x41FE
- [ ] 0x500F
- [ ] 0x47DA
- [ ] 0x4510
- [ ] 0x43ED
- [x] Nessuna delle altre risposte

Domanda 3 I record di attivazione:

- [ ] Nessuna delle altre risposte
- [ ] Sono necessari solo in presenza di funzioni di ordine superiore
- [ ] Sono allocati solo nello heap
- [ ] Devono essere esplicitamente allocati e deallocati dal codice del programma che li usa
- [x] Possono essere allocati sia sullo stack che sullo heap (in caso, per esempio, di funzioni di ordine superiore)

Domanda 4 β-riducendo $(\lambda d.((\lambda a.abc)(bc)))(\lambda y.xyz)$ si ottiene:

- [ ] La riduzione non termina
- [ ] Nessuna delle altre risposte
- [ ] $\lambda b.bba$
- [ ] $\lambda a.abc$
- [x] $bcbc$

```
int f4(int x)
{
  return 1 / x;
}

int f3(int a, int b)
{
  if (a > 1) return a; else return b;
}
```

Figure 1: Esempio di pseudocodice

Domanda 5 Un *interprete* di un linguaggio $\mathcal{L}$ scritto in un linguaggio $\mathcal{L}_O$ è:

- ☐ Un programma che trasforma un programma $P^{\mathcal{L}}$ (espresso nel linguaggio $\mathcal{L}$) in un programma $P^{\mathcal{L}_O}$ (espresso nel linguaggio $\mathcal{L}_O$) tale che per ogni input I si ha $P^{\mathcal{L}}(I) = P^{\mathcal{L}_O}(I)$
- ■ Nessuna delle altre risposte
- ☐ Una implementazione di macchine astratte indipendente dalla macchina fisica
- ☐ Un programma scritto nel linguaggio $\mathcal{L}$ che dato un input I produce lo stesso output generato dallo stesso programma scritto nel linguaggio $\mathcal{L}_O$
- ☐ L'implementazione di una macchina astratta scritta nel linguaggio $\mathcal{L}_O$, che capisce programmi scritti nel linguaggio $\mathcal{L}$

Domanda 6 L'*ambiente* (o *environment*) è:

- ■ Nessuna delle altre risposte
- ☐ Un insieme di associazioni (nome, valore) definite staticamente durante lo sviluppo di un programma
- ☐ L'insieme delle associazioni (variabile, valore) esistenti in uno specifico punto del programma ed in uno specifico momento durante l'esecuzione di un programma
- ☐ L'insieme dei valori che una variabile assume durante l'esecuzione di un programma
- ☐ Una lista di coppie (nome, tipo) che permette di accedere alle variabili di un programma

Domanda 7 Si consideri lo pseudo-codice di Figura 1. Qual'è il valore di ritorno di `f3(10, f4(0))` se i parametri sono passati *per valore*?

- ☐ 10
- ☐ Non è possibile dirlo senza conoscere il tipo di scope (statico o dinamico) utilizzato
- ☐ Non è possibile passare `f4(0)` per valore
- ■ Nessuna delle altre risposte
- ☐ 1

Domanda 8 In presenza di variabili modificabili:

- ☐ Nessuna delle altre risposte
- ☐ La valutazione del comando di assegnamento restituisce sempre un valore
- ☐ Il comando di assegnamento non ha effetti collaterali
- ■ Esistono un *Ambiente* che associa valori denotabili (fra cui le locazioni di memoria) a nomi ed una *Memoria* che associa locazioni di memoria a valori memorizzabili
- ☐ Non esistono valori denotabili

```
int i = 0;
int x[10];

int f2(int z)
{
  i = i / 2;
  return z - i;
}

int f1(void)
{
  int y;

  i = 4;
  x[0] = 1; x[1] = 0; x[2] = 2;
  x[3] = 4; x[4] = 6;
  y = f2(x[i]);

  return y + i;
}
```

Figure 2: Esempio di pseudocodice

Domanda 9 Si consideri lo pseudo-codice di Figura 2. Qual'è il valore di ritorno di `f1()` se i parametri sono passati *per valore*?

- ☐ 0
- ☐ 2
- ☐ Dipende dal tipo di scope (statico o dinamico) utilizzato
- ☐ 4
- ■ Nessuna delle altre risposte

Domanda 10 Dato il frammento di programma (espresso in pseudo-codice) contenuto in Figura 3, qual'è il valore di ritorno di `f1()`, assumendo scope dinamico?

- ☐ Non è possibile dirlo
- ☐ 5
- ■ 0
- ☐ Nessuna delle altre risposte
- ☐ -5

Domanda 11 β-riducendo $(\lambda n.\lambda f.\lambda x.f((nf)x))(\lambda f.\lambda x.f(f(f(fx))))$ si ottiene:

- ☐ Nessuna delle altre risposte
- ☐ L'espressione è irriducibile
- ☐ La riduzione non termina
- ☐ $\lambda f.\lambda x.ffffffx$
- ☐ fx
- ■ $\lambda f.\lambda x.f(f(f(f(fx))))$

Domanda 12 La *frammentazione interna* causa:

- ■ Uno spreco di memoria
- ☐ Il funzionamento non corretto di programmi che allocano memoria dinamicamente
- ☐ Un rallentamento rilevante nelle operazioni di allocazione della memoria
- ☐ L'impossibilità di allocare grandi blocchi di memoria anche se la memoria libera totale è sufficiente
- ☐ Nessuna delle altre risposte

```
int x, y, z;

void f3(void)
{
  x = 0;
  y = 5;
}

void f2(void)
{
  int y;

  f3();
  y = 0;
  z = 10;
}

int f1(void)
{
  int x;

  x = -5;
  y = 10;
  z = x + y;
  f2();

  return z - y - x;
}
```

Figure 3: Esempio di pseudocodice